Faculty of Computing

Report
Version 1.0

MSc Computer Science

A project management tool for FoC students projects

By
Lucas D Ruano-Garcia
07018295

September 2011

Executive summary

Project management is a key technique used by many when carrying out a project, as a consequence there are many tools available for a fee or for free. The use of these tools is wide spread in a professional and educational environment, however many students who carry out their projects at the university level find that these tools are not suitable for them for various reasons.

Common reasons given for project management tools not to be used by university level students are, usually that they think they do not need such a tool (due to bad practises when carrying out a project), the use of such a tool would be more of a waste of time than a help, the tool is too complex to use and would take too long to learn its features therefore and that the student believes they can carry out such activities with other tools that may not be fully suited or beneficial to them, such as spreadsheet software and word processing software.

The key factors to remember from the above are that, tools seem either overly complex or not useful for their particular application (a student project), therefore this project aims to overcome such objections and produce a tool aimed solely at students that can both help them perform project management techniques (specifically time management) successfully and with ease. This report details a project that shall create such a tool.

The tool successfully carries out the initial stated goals and performs well in the stated environment; however as with any project there is plenty of space for improvement in areas such as better handling project priorities and accounting for the use of resources in projects and dealing with them appropriately.

Acknowledgement

Many thanks go to all the people who have helped me in various levels of their capacity. Specifically I would like to thank Boris Cogan the supervisor who has provided the most help and guidance in structuring and carrying out this project as a whole. Also many thanks go to fellow students with whom ideas and understanding of the dissertation in general has been gained.

Table of Contents

1. Chapter 1 Introduction 5
1.1 Background of the Project 6
1.2 Problem domain 6
1.3 Business Requirements 6
1.3.1 Background, Business Opportunity, Customer Needs 6
1.3.2 Business Objectives and Success Criteria 7
1.3.3 Business Risks 8
2. Chapter 2 Project Management 9
2.1 Project statement of work 9
2.1.1 Product requirements 9
2.2 Project scope and statement 10
2.2.1 Project and product objectives 10
2.2.2 Product requirements and characteristics 11
2.2.3 Project requirements and deliverables 11
2.2.4 Project constraints 12
2.2.5 Project assumptions 12
2.2.6 Schedule milestones 13
2.2.7 Project configuration management requirements 14
2.3 Project life cycle model 14
2.3.1 Introduction 14
2.3.2 Project life cycle model 14
2.3.3 Model 15
2.4 Reason for choosing this methodology 16
2.4.1 Relation to the requirements of this project 16
2.4.2 Key advantages 16
2.4.3 Summary 17
2.5 Techniques used from methodology 17
2.5.1 Introduction 17
2.5.2 Analysis phase 17
2.5.3 Design phase 18
2.6 Languages used for the development of the tool 18
2.7 Tools used to develop the product 18
2.8 Project management plan 19
2.8.1 Work breakdown structure 20
2.8.2 Project schedule 21
3. Chapter 3 Vision of the Solution 23
3.1 Vision Statement 23
3.1.1 Detailed background of the problem 23
3.1.2 Derived key functions for the tool 25
3.1.3 Major Features 26
3.1.4 Assumptions and Dependencies 26
3.2 Scope and Limitations 27
3.2.1 Scope of Initial Release, Scope of Subsequent Releases 27
3.3 Business Context 27

3.3.1 Project Priorities 27
4. Chapter 4 Similar software developed before: Literature survey 29
4.1 Comparison criteria 29
4.2 Existing software 30
4.2.1 Microsoft Project 31
4.2.2 ConceptDraw Project 33
4.2.3 FastTrack Schedule 34
4.2.4 GanttProject 35
4.3 Conclusion 37
Chapter 5 Analysis 38
4.4 Background of the problem 38
4.4.1 What are *project management tools*? 39
4.4.2 What does a *FoC student project* consist of? 40
4.5 User Requirements 40
4.5.1 Case Study 40
4.6 Functional requirements defined 41
4.6.1 Add a task 41
4.6.2 Modify a task 41
4.6.3 Delete a task 42
4.6.4 View a task 42
4.6.5 Prioritise tasks 42
4.7 Non-Functional requirement defined 42
4.7.1 Configuration management 43
4.7.2 Platform compatibility 43
4.7.3 Maintainability 43
4.7.4 Usability 44
4.8 Use Case model 44
4.8.1 Description 44
4.8.2 Diagram: use case 45
4.9 High level use case description 45
4.9.1 Add a task 45
4.9.2 View a task 46
4.9.3 Delete a task 46
4.9.4 Prioritise a task 46
4.10 Expanded use case description 46
4.10.1 Add a task 46
4.10.2 View a task 47
4.10.3 Delete a task 48
4.10.4 Prioritise a task 49
4.11 Activity diagrams 49
4.11.1 Description 49
4.11.2 Diagram: Add a task 50
4.11.3 Diagram: View a task 51
4.11.4 Diagram: Delete a task 52
4.11.5 Diagram: Prioritise a task 53
4.12 Collaboration diagram 54

4.12.1 Description 54
4.12.2 Diagram: Add a task 54
4.12.3 Diagram: View a task 56
4.12.4 Diagram: Delete a task 57
4.12.5 Diagram: Prioritise a task 58
5. Chapter 6 Design 59
5.1 Class diagram 59
5.1.1 Description 59
5.1.2 Diagram 60
5.2 Data dictionary 60
5.3 User Interface design 62
5.3.1 Story board 62
6. Chapter 7 Construction and testing 64
6.1 Construction 64
6.1.1 ProjectTask 64
6.1.2 ProjectController 65
6.1.3 ProjectFrame 66
6.2 Testing 66
6.2.1 Purpose of testing 66
6.2.2 Testing Strategy 67
6.3 Test Case's 67
6.4 User Interface Testing 72
6.4.1 Consistency testing 72
6.4.2 Button testing 73
6.4.3 Menu testing 74
6.5 Testing Review 75
6.5.1 Pass & Failure rate 75
6.5.2 Errors and Solutions 75
6.5.3 Conclusion 76
7. Chapter 8 Conclusions and evaluation 77
7.1 Conclusion 77
7.2 Evaluation 78
8. References 79
9. Bibliography 81
10. Appendix 83

Abbreviations

Abbreviations of certain words have been used in this report to both save time and space, a list of these abbreviations and their meaning is shown below in the following table.

Abbreviation	Definition
FoC	Faculty of Computing
V	Version(s)
RUP	Rational Unified Process
USDP	Unified Software Development Process
UML	Unified Modelling Language
INC	Include(s)/(ing)
HCI	Human Computer Interface
WBS	Work Breakdown Structure

Interchangeable terms

Several terms have been used in the report some of which refer to the same thing; therefore for clarities sake the following illustrates such related terms.

- Software, Application or Tool

  These terms and other similar such terms are used interchangeably to aid the discussion of projects on many levels.

- Activities or Workflow

  These terms and other similar such terms are also used interchangeably to discuss the progress of the project.

1. Chapter 1 Introduction

Introduction

This project is concerned with allowing FoC students doing their final year projects to properly and easily deal with the project management aspect of each, by creating an intuitive and focused tool to carry out such activities following proper software engineering practises. This project shall produce two deliverables once the project is closed, those deliverables shall be this project report and a project management tool.

A brief list of aims and objectives for this project is given below:

- Allow students to properly deal with the many tasks in a project
- Have the facility to keep track of a project by tasks
- Have the ability to prioritise such tasks

A rigid structure shall be used to produce this report which shall allow for relevant information to be gained methodically which can be used in later parts of the report therefore following appropriate software engineering practises. Notable chapters in the report that are extremely important to the development of the project management tool are the analysis chapter through to the testing chapter.

The structure of the report that shall be followed (as mentioned above) shall take into consideration both international standards and faculty standards. Below is a list of chapter names illustrating the report structure that shall be followed.

- Project Management chapter
- Literature survey chapter
- Methodology chapter
- Analysis chapter
- Design chapter
- Construction & Testing chapter
- Conclusions & Evaluation chapter

- User manual
- Appendix
- Reference section

This chapter shall mainly be concerned with introducing key concepts of this project while giving a background to the project as a whole and purpose/need of general tools relating to the one that shall be developed. The problem domain shall also be briefly discussed in this chapter.

1.1 Background of the Project

1.2 Problem domain

Within the FoC currently there are very few tools that aid students carrying out projects to efficiently and effectively plan the project management aspect of each. Most or at least some of the computers at the faculty come with Microsoft Project already installed as the main software for planning and carrying out projects, however since many students have a low familiarity with the software and since it is so broad and general it can seem daunting or not useful for them. It is for these reasons among others that Microsoft Project is not used as a means of planning student projects.

Since there is a great need for the proper management of projects an appropriate and accessible tool for students would be highly beneficial in not only managing the projects but also in maintaining a good discipline when dealing with other similar endeavours.

1.3 Business Requirements

1.3.1 Background, Business Opportunity, Customer Needs

Currently at the FoC, students carrying out projects will go through several key steps during the project, these will most commonly be carrying out research on a topic area, planning then developing the projects products. Although there are many tools available to students carrying out their projects, they will usually not make use of them due to many different reasons. Some times a tool is hard to learn, is not applicable to their smaller sized projects or is not easily available to students. Due to these many reasons

most planning of projects are usually done manually; most commonly by creating a Gantt chart with preconceived milestones (some times using a spreadsheet but most commonly using any available word processor), this can lead to several project issues that could have a serious impact either on the project or the key planning skills of the student. Such issues that could occur, would be that since a plan is likely to change throughout the project, the manual tool (in this case it could be a self made Gantt chart) would have to be completely redone, which would take more time than it should or at worst, it could lead to laziness by the student who would see the planning as a chore and therefore avoid putting much effort into it in later projects.

For these reasons a suitable tool is needed not only to help facilitate the proper management of a project but also to maintain good habits for a student when dealing with future projects. Therefore for such a tool to be considered suitable several things would need to be taken into consideration. In general these things can be classified as such:

- The tool has to be able to deal with common project management techniques
- The tool has to be suitable for simplistic to semi-complex projects at the level of a post graduate dissertation
- The tool should enforce proper planning of projects through its use
- The tool has to be intuitive and accessible to students of any level of knowledge

If such a tool could provide such an experience to students then the benefits would be huge both practically (in helping deal with the project more efficiently), professionally (since projects are common to the work place and therefore consequently so is the need for project management) and personally (since better project management can also help in managing daily activities in life better too).

1.3.2 Business Objectives and Success Criteria

Several things are needed before carrying out this project, namely they are the actual objectives towards which this work shall be directed (which shall then provide a basis for a deeper understanding of will be required of this project) and how each objective shall

be judged to be successful or not. Below is a list of these objectives along with their success criteria, both of which shall for the basis for the following chapters.

BO-1: Track a student project by day, month and year.
Scale: Dependant on the particular project.
Meter: Record the start of a project and record the end of each task within the project then record the closing of the project.
Goal: A clear idea of the time spent on tasks and projects will be gained.
Fail: Forgetting to record the start or end of a task and or project.

BO-2: Maintain an up to date list of ongoing tasks within the project
Scale: Dependant on the particular project.
Meter: Check all ongoing tasks on start up noting any tasks that should be there but are not listed
Goal: All ongoing tasks shall be frequently recorded.
Fail: User might fail to add a deadline to a task or add a wrong deadline therefore making it not show up in the list.

1.3.3 Business Risks

As with any project there may be several risks associated with the final product that might hinder it from fulfilling its intended purpose. These risks can have different likely hoods of occurring and also different levels of impact on the both the product and the project. Below is a list of all risks associated with this project along with their probability of occurring and how severe the impact of each might be.

RI-1: The student might not find the tool intuitive and too confusing therefore avoiding using it. (Probability = Unlikely. Impact = Severe)
RI-2: The student might find the tool to be too cumbersome and not an efficient use of their time. (Probability = Likely. Impact = Severe)

2. Chapter 2 Project Management

Introduction

The following chapter contains details of how the project shall be managed as well as discussing the project as a whole and the exact elements that shall make up this project in particular. A detailed explanation of what will be carried out and how it will be carried out along with an illustration of how the project should progress shall also be mentioned in this chapter.

2.1 Project statement of work

2.1.1 Product requirements

As mentioned in the previous chapter for a project management tool for students to be successful several things must be true, these things form the basis of the product requirements of this project and are what shall dictate the functionality of the tool later on in development.

There are several stakeholders in this project that shall have an influence on how the products shall take shape and therefore be a source of requirements. Those stakeholders shall be the creator or the tool and all project products, students who will use the tool and both the supervisor to this project and the module leader.

1. The product shall allow tasks to be added, deleted and modified
2. Each tasks in a project shall have the following details attached to them: task name, description of task, dependencies, start date and time, end date and time, duration, any needed resources and if the task is mandatory or optional
3. The product shall maintain a list of ongoing tasks (not yet completed), overdue tasks and completed tasks
4. The product shall allow a project to be viewed chronologically in the following way: a view of each day within a month and a view of each task within a day
5. The product shall allow the tasks within a project to be viewed according to dependencies

6. The product shall allow tasks to be broken down into sub tasks.
7. The product shall attach a priority to each task determining the order that tasks should be carried out (according to their place in the project, their dependencies and deadlines).

2.2 **Project scope and statement**

2.2.1 Project and product objectives

2.2.1.1 Project objectives

For this project to be successful several things will be needed, in most cases this is true of most projects but in some cases these are unique to this particular type of project. These things can be seen as essential objectives to the project, and below they shall be listed in order of priority and explained.

1. The Project management, Vision of the solution and Literature survey chapters of the project must be completed by the 28th June
2. The project shall be closed delivered on the deadline of 02 September
3. The project must maintain a high level of quality throughout its life cycle, this will shall be defined by the following parameters:
    - Setting and measuring milestones and seeing how many are met and how many are not met
    - Meeting the final deadline
    - Reaching a level of project maturity of 80% (or meeting 5 out of 8 requirements)
4. The project shall be planned so that it can make use of all resources allocated to it without needing any more than what is available

2.2.1.2 Product objectives

As the project needs certain things to allow its successful completion the product too must also have a set of objectives to follow so as to allow it to be successful. Below is a list of these product objectives in order of priority along with their explanations.

1. By carrying out the product with quality standards in mind, all products shall have little to no critical bugs and errors
2. The tool shall satisfy at least two thirds of the requirements
3. The product will be maintainable and extendable by providing a user manual and any other suitable documentation
4. The product shall meet each milestone and be delivered on the deadline 02 September

2.2.2 Product requirements and characteristics

The product shall be used by students too help with their project management and therefore allow for certain functions to be carried out easily and intuitively. This tool shall take the form of an executable program run from the user's computer and able to run without installation.

2.2.3 Project requirements and deliverables

The project will need to ensure certain things are true for it to be carried out successfully, these requirements are as follows;

- The main report for the project shall be compile then delivered once the project is closed meeting the specified deadlines
- Regular contact and meetings with the assigned supervisor shall be observed with a logbook of each meeting and record of each instance of contact that is not considered a meeting
- Three products shall result from this project and they shall be, the tool, a user manual for the tool and supporting report for the project, all of which shall be complete and accurate according to the defined quality in the appropriate section of this report.

Knowing the requirements a list of deliverables can be picked out for this project and they shall be as follows:

- Logbook
- User manual
- Project report
- Project management tool

2.2.4 Project constraints

This project may face several constraints that would jeopardise the satisfactory completion of the project, below is a list of such constraints and how and what they may affect the project and product.

Constraint	How likely is it and how severe would its impact be	How would you reduce the probability of a risk becoming a reality	How would they be handled if it became a reality
Schedule Time	Likely/High	Proper planning beforehand	Go over plans and readjust them
Resources	Unlikely/Low	Choose a project that doesn't require additional resources	Change the part that would require the additional resources
Current skill	Moderately likely/Medium	Understand area and research needed skills	Do further reading and research
Current knowledge	Moderately likely/Low	Plenty of background reading	Do further reading and research

As can be seen from the table above there are quite a few general risks that could affect the project as a whole and each of them would have a different impact on the completion of this endeavour. The highest risk to the project (as can be seen in the table) is that of running out of time, in this case what this means is not finishing the project on time, this is something that can be hard to predict and can have disastrous results if it does happen which is why proper planning and some flexibility in the plan would help.

Since this project shall use readily available resources and not require anything currently unavailable, budget shall not be considered for this project.

2.2.5 Project assumptions

Assumptions can have the potential to cause problems in a project due to the reliance on resources that might not be available when expected or to a smaller extent due to

complacency about the project environment. Some project assumptions have been identified and are listed below.

The following assumptions might impact the project's success if false:

- The project plan has been carried out well and needs no adjusting
- Resources will be available (hardware and software)
- Time scheduled shall be followed
- No further skills need to be learnt or revised

Having identified these assumptions attempts to avoid them shall be taken so as not to jeopardise the project.

2.2.6 Schedule milestones

As with any project there are several milestones that should be taken note of since they can be used to measure the progression of the project at various stages. Since tracking the progression of the project is a key practise in ensuring appropriate project management technique, several milestones have been identified and are outlined and numbered below.

1. Milestone 1- Project initiation
    - Initial research
2. Milestone 2- Definition of problem
    - Vision of solution chapter
3. Milestone 3- Project management
    - Project management chapter
4. Milestone 4- Analysis
    - Draft of analysis chapter
5. Milestone 5- Product design
    - Draft of the design chapter
6. Milestone 6– Construction and testing
    - Construction of product and relevant report chapter
    - Testing of product and relevant report chapter
7. Milestone 8- Project conclusion
    - Final draft of the project report

- Hand in of user manual
- Hand in of logbook
- Hand in of software artefact

2.2.7 Project configuration management requirements

Since two people (author and supervisor of this project) shall be using the same document (this report) there is a definite need for some form of configuration management, this shall also ensure good practises and that the project as a whole has been carried out effectively. There are two main areas of configuration management that shall be paid attention to and those are as follows:

- Version control of both the report and product shall be observed
  - Predefined naming scheme for work products (for the report the following shall be used: *(initials of person who last edited document) (date of last edit) module code student ID.doc(x)*)
  - Whenever a new version of a work product is made, the older version shall be saved and archived and the new version worked on
- A level of change control shall be used for all work products
  - Clear identification of changes made to the project report and tool

2.3 **Project life cycle model**

2.3.1 Introduction

From the scope definition and statement of work detailed above a life cycle model for the project can be derived. Since this is a student project the product is not intended to out last the project therefore both models have been combined into one.

2.3.2 Project life cycle model

The model shall be described in terms of phases with explanations for each one and the activities carried out in each then the model shall be presented with further explanations. The model shall be heavily influenced by RUP/USDP following an incremental/iterative approach therefore each phase can be carried out more than once producing deliverables within each increment.

Project Planning

- This is the phase that initiates the project opening the report and belongs to the project lifecycle. It is in this phase that the aim of the project in general is discovered as well as its scope and project planning is made, with project requirements being fleshed out.

Analysis

- Within this phase the main focus shall be gathering relevant information to form the requirements of the product, then evaluating them and ordering them in terms of priority. Therefore this part marks the beginning of the product lifecycle.

Design

- This is the next phase in the product lifecycle where the main focus is designing the product using diagrams and giving relevant explanations.

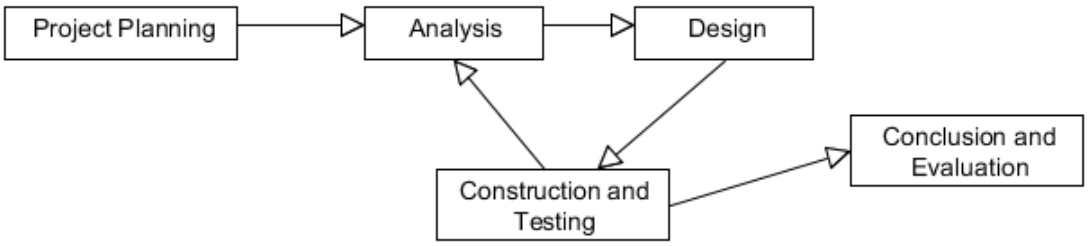
Construction/Testing

- The last part of the product lifecycle is mainly concerned with actually making the product and testing it. With this part done the product lifecycle is complete unless any other previous phase needs revising.

Conclusion and Evaluation

- The main focus of this phase is to produce a reflective point of view of the product and project as a whole noting where things could be improved. This marks the end of the project lifecycle unless any of the previous phases need revising and the closing of this report.

2.3.3 Model



Each box represents a phase mentioned above as well as an increment of the product and each arrow represents the flow of work. Despite all phases being iterative the phases seen in the middle (Analysis, Design and Construction and Testing) are the ones which shall be the most iterative. Each iteration of the above phases shall be carried out to produce increments that will come to form the final product and relevant chapters consequently leading the project as a whole to its completion.

2.4 **Reason for choosing this methodology**

2.4.1 Relation to the requirements of this project

By bearing in mind the requirements and their nature the appropriate methodology can be chosen, in this case many things have been taken into consideration so that the chosen methodology can be used for this project. The relationship between the requirements and this project and how they relate to the choice of methodology is detailed below.

A key factor to bear in mind is the size of the project as a whole, which is relatively small (therefore having few requirements) in nature due to the fact that it is a student project with many constraints. Therefore a suitable methodology would be needed that can deal with the simplicity and size (in comparison to industrial projects), meaning waterfall methodologies or similar would not be ideal. The fact that the products can be developed incrementally and iteratively means that a methodology that can support such an approach is needed.

2.4.2 Key advantages

There are many key advantages to using the chosen methodology for this project, the most obvious of which is the fact that one of the main products that shall be produced shall be the report, therefore a methodology that is heavyweight is ideal for this project (RUP/USDP meeting that criteria). As mentioned above an incremental/iterative approach is to be taken by this project therefore the chosen methodology can properly support the nature of this approach. Another key point is that since the requirements

might change the methodology will have to be flexible in this area, while maintaining the project on schedule, something that this methodology can cater for.

2.4.3 Summary

Overall the benefits of using the chosen methodology in this project are clear and can be seen to tightly fit in with the very nature of the work that is carried out. For these reasons, other methodologies have been ruled out and the chosen methodology (influenced by RUP/USDP) shall be used.

2.5 **Techniques used from methodology**

2.5.1 Introduction

The basis for the chosen methodology (RUP/USDP) contains many techniques for carrying out a project those that shall be used in this project shall be detailed below in their relevant phases. It should be noted though that RUP/USDP do not give much guidance after the design phase therefore general project management techniques shall be used. Since RUP/USDP give UML as the preferred method for carrying out a project these shall be the techniques discussed below.

2.5.2 Analysis phase

Once the requirements have been elicited from the key project stakeholders (both of which shall be identified), then a use case diagram shall be developed to organise the requirements and gain an understanding of the final product and any interactions that the users will have with the application. Once the use case diagram has been made each use case shall be extracted and both a high level use case description and an expanded use case description shall be produced to better understand how the requirements relate to functionality being designed.

With the use cases and their descriptions done, the details contained in them shall be used to create two additional diagrams. One of these diagrams shall be the activity diagram which shall help visually understand the steps needed by the user and the tool to carry out

certain functions. The other technique that shall be used would be collaboration diagrams, these shall enable a better understanding of object interactions in the program, as well as providing a lead into the design stage.

2.5.3 Design phase

The diagrams and information gained from the previous phase shall be fed into this phase and used to create more diagrams and information. Only a few UML techniques shall be used at this stage all of which will take the information and diagrams produced from the previous stage to help produce a clear design of the final product. The first diagram that shall be produced at this stage shall be the class diagram which uses the classes from the collaboration diagram to better understand their relationship.

Two more techniques shall be used from the chosen methodology at this stage, they shall be a state charts and package diagrams so that a full view of what needs to be produced is laid out and clear. These shall be the main UML techniques as dictated by RUP/USDP. Many other things shall be carried out in these stages but they shall all be part of the detailed techniques one way or another.

2.6 **Languages used for the development of the tool**

Due to the nature of this project and the tool being designed only one development language shall be used and that is Java (the latest version). The reason for choosing this language is due to the fact that the tool shall be a desktop application and useable on as many platforms as possible. Also the choice of this development language relates to the knowledge of the implementer and classes already available.

2.7 **Tools used to develop the product**

Certain tools shall be used during the course of the development of the application, these shall either be to build the actual tool or compile the report and any other necessary menial tasks. These tools are given below with a short explanation of how they shall be used.

- Microsoft Word 2003 – This shall be used to compose the project report.

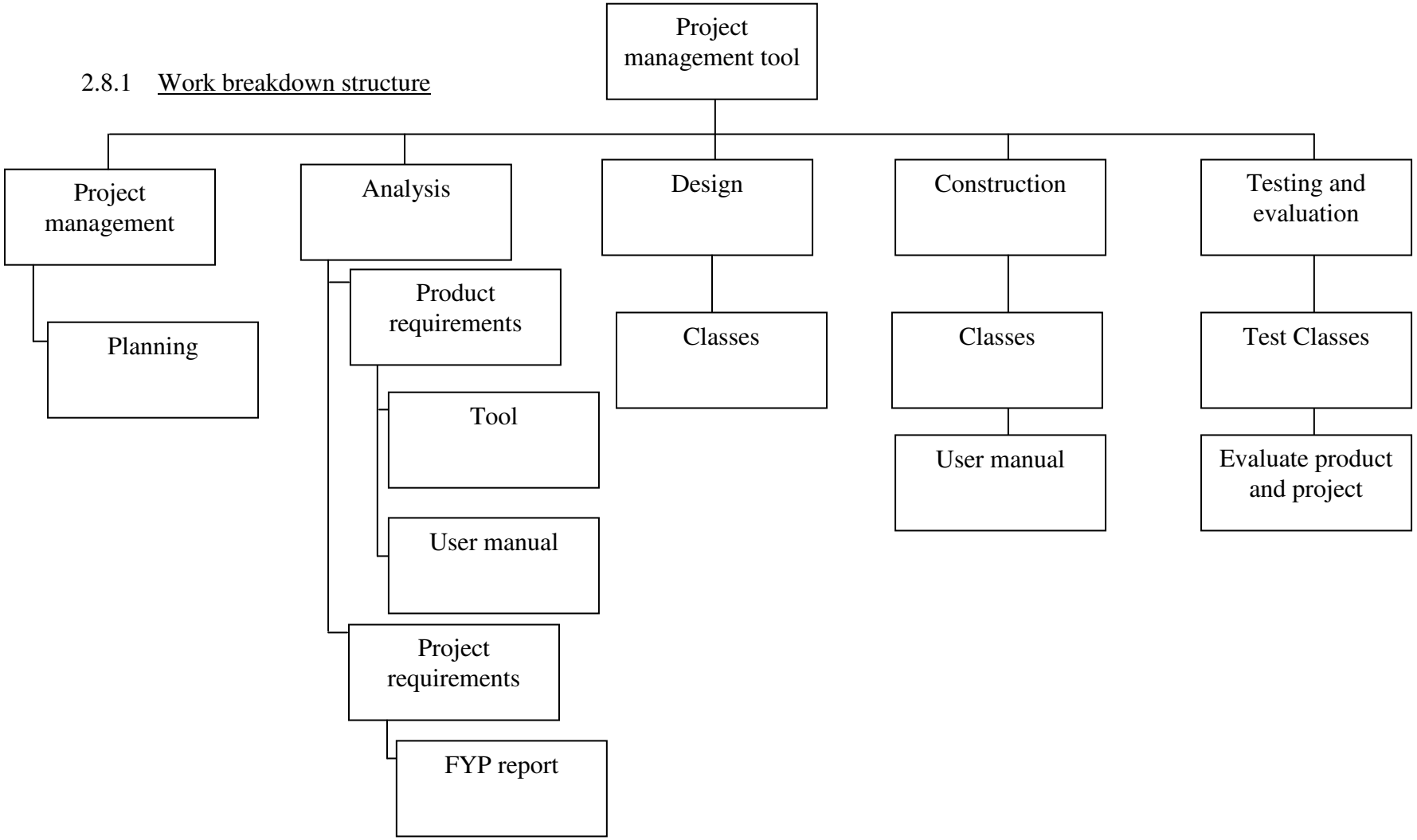
- Microsoft Excel 2003 - This shall be used to produce graphs for the report on the project side of the development.
- Microsoft Project 2007 - This shall be used to produce Gantt charts and manage the project.
- NetBeans 6.9 - This shall be used to write the source code for the classes that shall make up the finished application.
- GIMP 2.6 - This shall be used to manipulate any images that might be used.

From the above tools the ones that shall be used that most shall be Microsoft Word and NetBeans. Additionally although it is not a tool by definition a previous project that has been worked on shall be used to base key ideas and background to this project.

2.8 **Project management plan**

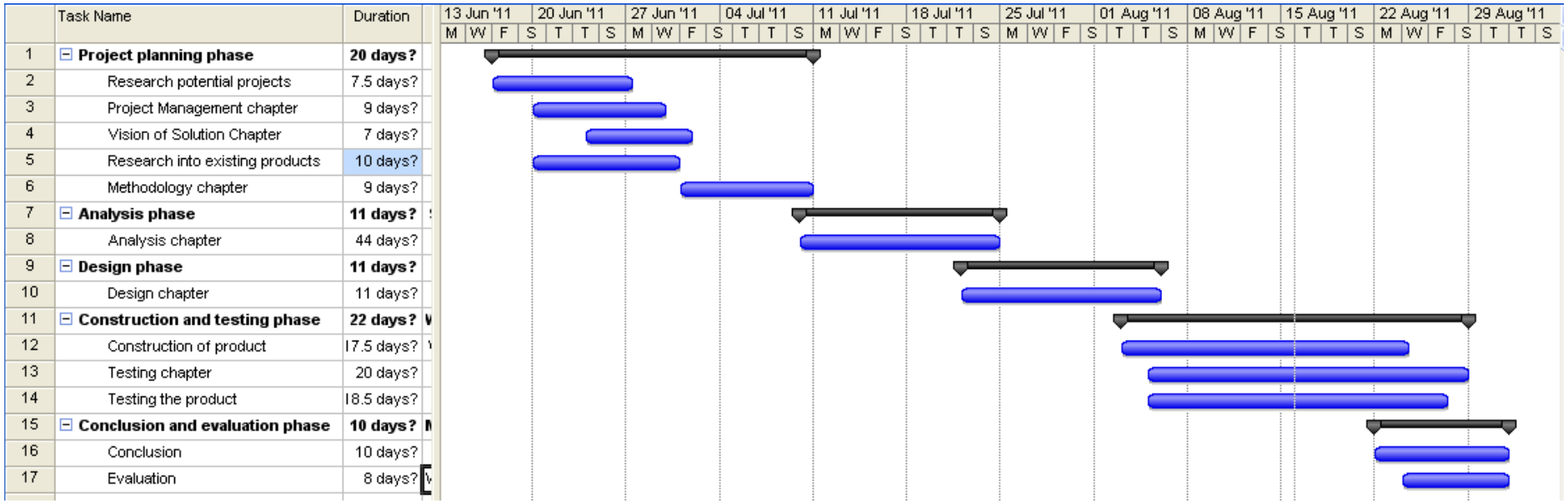
Using the milestones from above and gained information previously obtained, a plan of how the project shall be managed can be made. The first step is to come up with a WBS to organise the work to be carried out logically, then using the milestones and the newly created WBS a project schedule can be defined.

2.8.1 Work breakdown structure



2.8.2 Project schedule

What follows is the project schedule as depicted in a Gantt chart showing each phase and the activities that shall be carried out within them along with the duration of each and graphically the start and ending of each.



	Task Name	Duration
1	**Project planning phase**	**20 days?**
2	Research potential projects	7.5 days?
3	Project Management chapter	9 days?
4	Vision of Solution Chapter	7 days?
5	Research into existing products	10 days?
6	Methodology chapter	9 days?
7	**Analysis phase**	**11 days?**
8	Analysis chapter	44 days?
9	**Design phase**	**11 days?**
10	Design chapter	11 days?
11	**Construction and testing phase**	**22 days?**
12	Construction of product	17.5 days?
13	Testing chapter	20 days?
14	Testing the product	18.5 days?
15	**Conclusion and evaluation phase**	**10 days?**
16	Conclusion	10 days?
17	Evaluation	8 days?

2.8.2.1 Dependencies

In several places tasks cross over, this indicates that one task does not need to be fully completed for the next task to be started, however there may be instances where a task maybe dependant on another. This can be shown by one task not being started until the other is finished. In most cases this is due to the fact that information gathered in one task is needed by another task e.g. the construction of the database can only happen after the analysis chapter is completed and a substantial amount of T7 the Design chapter is completed. Mostly though for this project a phase can start after key activities in the previous phase have taken place and although

not shown each phase can be iterated (and therefore potentially take longer than shown above), when this is the case the start date is taken as certain and the duration as a recommendation.

3. Chapter 3 Vision of the Solution

Introduction

This chapter is concerned with establishing a clear vision of solution using all information gathered from the previous chapters. To achieve this, a detailed breakdown of what happens in a student project shall be established as well as who is involved at what stage, and what documents shall be produced as along with their structure.

3.1 Vision Statement

3.1.1 Detailed background of the problem

3.1.1.1 Project management knowledge areas

Project management can be seen as having nine knowledge areas where certain activities are carried out in each forming documents that can seen as deliverables for each. PMBOK 2004 [1] details all project management knowledge areas with very detailed explanations of what activities are carried out in each and the documents required and produced. These knowledge areas from PMBOK 2004 [1] are listed below with details of the knowledge area that shall be focused on for this project.

- Project Integration Management.
- Project Scope Management.
- ***"Project Time Management**, describes the processes concerning the timely completion of the project. It consists of the Activity Definition, Activity Sequencing, Activity Resource Estimating, Activity Duration Estimating, Schedule Development, and Schedule Control project management processes."*
- Project Cost Management.
- Project Quality Management.
- Project Human Resource Management.
- Project Communications Management.
- Project Risk Management.
- Project Procurement Management.

3.1.1.2 Project time management activities

Throughout a student project many activities shall take place in each of the above mentioned knowledge areas, one of the most notable set of activities can be found in the project time management knowledge area. Therefore with an understanding of the knowledge areas in project management and specifically what project time management entails and consists of, a list of activities can be established. These activities have also been detailed in PMBOK 2004 [1] with what each of these activities involve and are listed below.

- Activity Definition- identifying the specific schedule activities that need to be performed to produce the various project deliverables.
- Activity Sequencing- identifying and documenting dependencies among schedule activities.
- Activity Resource Estimating- estimating the type and quantities of resources required to perform each schedule activity.
- Activity Duration Estimating- estimating the number of work periods that will be needed to complete individual schedule activities.
- **Schedule Development**- analyzing activity sequences, durations, resource requirements, and schedule constraints to create the project schedule.
- **Schedule Control**- controlling changes to the project schedule.

3.1.1.3 Key project time management activities description

PMBOK 2004 [1] gives an in depth explanation of each activity with descriptions of each input, the tools and techniques used and the resulting outputs for the given activities. This project shall focus primarily on the Schedule Development and Schedule Control activities.

Schedule Development

According to PMBOK 2004 [1] this activity is said to help find out when the start and finish dates are for activities within the project possibly a review or revision of duration estimates and resource estimates so that the project schedule can be created. This can be used as a baseline for which tracking of the progress can be done.

A key technique/tool in relation to the tool being developed is the schedule model which according to PMBOK 2004 [1] uses schedule data and information which get compiled into the schedule model. There are several key outputs of the schedule development activity, one of which is the project schedule which according to PMBOK 2004 [1], "includes at least a planned start date and planned finish date for each schedule activity." The other is schedule model data which contains various supporting data those being, "schedule milestones, schedule activities, activity attributes and documentation of all identified assumptions and constraints." PMBOK 2004 [1].

Schedule Control

A tool to this activity is the project management software which PMBOK 2004 [1] states as such, "Project management software for scheduling provides the ability to track planned dates versus actual dates, and to forecast the effects of project schedule changes, real which makes it a useful tool for schedule control."

3.1.2 Derived key functions for the tool

With a firm understanding of the project management key principles along with their activities and how they work, a set of key functions can be derived. Below is a break down of the derived key functions and their sub functions that users might want or need from the tool that shall be produced.

Group of functions

- Create the project schedule
- Track the projects progress

Individual functions

- Create the project schedule
    - Store tasks
    - Manage tasks

- Track the projects progress

- Track planned dates versus actual dates
- Track schedule milestones

Sub-functions

- Store tasks
  - Make, modify and delete a task
  - Set task details

3.1.3 Major Features

From the above derived functions a list of features of the tool can be obtained. Below is such this list with each feature and sub feature labelled for reference later on in the report.

FE-1: Create the project schedule

FE-1.1: Store tasks

FE-1.2: Manage tasks

FE-2: Track planned dates versus actual dates

FE-3: Track schedule milestones

3.1.4 Assumptions and Dependencies

Assumptions can be made that might be to the detriment of the tool, therefore it is best to identify as many as possible as early as possible. Noting dependencies as early as possible that the tool might have is also very useful. Below both assumptions and dependencies are listed and labelled as appropriate.

AS-1: The student will provide the correct details for each task (such as deadlines etc.)

AS-2: The tool shall be able used regularly

AS-3: The tool shall be used along with other tools and techniques

DE-1: The computer from which the tool is run will have the minimum system requirements

DE-2: The student has project management knowledge

3.2 **Scope and Limitations**

3.2.1 Scope of Initial Release, Scope of Subsequent Releases

Certain features shall be implemented before others which, this might lead to one release of the tool having either partial, full or no implemented features. In subsequent releases those features that were either partial implementations or not implemented at all shall be fully implemented. A table is shown below using the features with their respective labels given above and the scope of release for each feature.

Feature	Release 1	Release 2
FE-1	Partially implemented	Fully implemented
FE-1.1	Fully implemented	-
FE-1.2	Partially implemented	Fully implemented
FE-2	Fully implemented	-
FE-3	Not implemented	Fully implemented

Features that have been fully implemented in the first release will continue to be fully implemented on the second release therefore a dash (-) is used in the table above to show this and avoid repeating information.

3.3 **Business Context**

3.3.1 Project Priorities

Below a table is given showing the project priorities with a small explanation.

Dimension	**Driver (state objective)**	**Constraint (state limits)**	**Degree of Freedom (state allowable range)**
Schedule		Final deadline can not be over run	None
Features			At least half of the features must work by version one, and all by version two

Quality			High quality on first version, highest quality on second version
Staff	Maximum team size is 1 developer and tester		
Cost		Budget can not be over run	

4. Chapter 4 Similar software developed before: Literature survey

Introduction

A CASE tool can take many forms but most if not all those who use them will do so for a particular purpose, those being mainly for the analysis and design of both a project and its products according to a predefined life cycle. The European Research Consortium for Informatics and Mathematics (ERCIM) defines the term CASE tool as such:

> "A software tool that helps software designers and developers specify, generate and maintain some or all of the software components of an application."[2]

While the Association for Project Management (APM) defines project management as such:

> "Project management is the process by which projects are defined, planned, monitored, controlled and delivered such that the agreed benefits are realised. Projects are unique, transient endeavours undertaken to achieve a desired outcome."[3]

Therefore a project management CASE tool would have to match both descriptions to be successful.

It is highly unlikely that some kind of similar software to the tool being developed is not already on the market or available. This makes it very useful to research such software to both gain an idea of their features and the expectations that the user of these software's might have. For these very reasons this chapter shall contain the most similar (according to a predefined comparison criteria) software's found and documented.

4.1 Comparison criteria

To be able to properly compare software similar to the one that shall be developed specific criteria are needed so that those software's which are not similar, will not be considered and consequently included in the comparison. A good way to devise comparison criteria is to decide on the features that the designed software shall have and other functional aspects.

As mentioned above the key features of the tool shall be as follows:

- Allow tasks to be added, modified and deleted
- Track a task by various properties
- Track the whole project
- Allow different views of the project

Other non-functional aspects of the tool shall be:

- Easy to use (measured by how long it takes for someone who has not used the tool before to be able to carry out the tasks that they need).
- Minimalist user interface (this shall be seen as the UI having only the functions needed to carry out the tasks and nothing more)

Therefore when comparing other software these features shall be looked at to determine whether the software is similar or not to the tool being designed.

4.2 **Existing software**

Currently there are plenty of tools for managing projects; one of the most known tools is that which is provided by Microsoft separately from their Office suit, Microsoft Project. This particular tool has many features which might be suitable for university students doing their dissertations, there are however many more project management tools that could fulfil such a role from many other companies. Tools that shall be looked at apart from Microsoft Project shall be as follows:

- ConceptDraw Project
- FastTrack Schedule
- GanttProject

It must be noted that most of the currently available tools for project management are not specifically aimed at students who are doing their dissertations at university, therefore if they were to use such tools they might find them overly complex for their particular project, J. Paynter, notes in a study carried out to discuss Software engineering project management, estimation and metrics that, “The participants considered that the use of project management tools had a high learning curve...”[4] and notes that as a result of this, “...simpler approaches should be used where possible to meet the learning objectives.” [4] From this observation it is clear that the current tools out there are not suitable for students.

4.2.1 Microsoft Project

4.2.1.1 Description

Microsoft Project (MP) is Microsoft's solution to project management as part of the Microsoft Office suite, being available in two different versions; Microsoft Project Professional and Microsoft Project Server. As the Microsoft website states,

> *"Microsoft® Project Professional 2010 delivers a project management system with powerful, visually enhanced ways to effectively manage a wide range of projects and programs. From meeting crucial deadlines to selecting the right resources and empowering your teams, Project Professional 2010 helps project management professionals by offering easier and more intuitive experiences to be more productive and realize amazing results."[5]*

4.2.1.2 Features

MP comes with various features to facilitate project management. The following is a list of the key features of this tool that relate to the tool being designed.

- Gantt chart default view
- Use many different colours and text formatting for text and images
- Allow different views of the project, such as:
  - Calendar
  - Gantt Chart
  - Network diagram
  - Task usage
  - Tracking Gantt
- Handle resources for a project
- Track a task by the following:
  - Task name
  - Duration
  - Start
  - Finish
  - Predecessors
  - Resource names

4.2.1.3 Pros

Of the many features that MP contains, some stand out more than others that maybe be both seen as useful for managing a project but also useful when designing the tool for this project. These key useful features are listed below:

- Familiar layout for those that have used other Office programs making it easier to learn how to use.
- Ability to create many different charts and other useful tools, such as a PERT chart
- A project management plan can be easily set up from the beginning
- Can calculate many things (such as deadline and time to do tasks) automatically given certain information

4.2.1.4 Cons

Although there are many positive points, MP also has many negative points that could be seen to cause problems when planning a project or as things that the tool being designed may improve on. These things are listed below:

- Not easy to get hold of (in comparison to other Office programs), since it is sold separately from the main Office programs
- Platform dependent- only available for the Microsoft Windows platform. [5]
- Huge program that takes up lots of memory to install
- Can be overwhelming due to the large amount of functions and options

4.2.1.5 Usefulness to this project

To summarise Microsoft Project is a fully featured project management tool that can be used optimally for projects of medium to large sizes with the integration of other Office programs it has many advantages. However because it is so fully featured and large it can cause problems both for installing on a new system but also because of its sheer quantity of options and functions, this is quite significant in the fact that it can discourage its use on smaller scale projects and for those that are not used to using it. Therefore due to its size and complexity it is not suited to student projects, however its key features are useful to take not of when developing a tool that is suitable for students.

4.2.2 ConceptDraw Project

4.2.2.1 Description

ConceptDraw Project (CDP) is another fully featured project management tool that is available for both the Microsoft Windows platform and the Mac OS X platform. On the website the program is described thus:

> "*ConceptDraw PROJECT contains an extensive tool set to help project managers. The rich data visualization capability that is provided by ConceptDraw products builds project dashboards, one-click reports, multi-project views, Gantt charts, and resource views. The rich visual data presentation supports important project management tasks such as critical planning and change management.*" [6]

4.2.2.2 Features

As with MP, CDP has many features that deal with project management as a whole rather than a specific area, however it also offers a default view of the project by way of a Gantt chart. A list of the key features offered by this program is given below:

- Traces tasks to their requirements documents
- Tracks the project by its documentation
- Integrates with other software
- Allows tracking and reviewing milestones
- Provides different views of a project such as, Gantt charts and resource views [6]

4.2.2.3 Pros

Although the features are quite similar to those found in other project management programs, there are a few features which stand out among them that can be seen as useful to this project. These features are:

- Resources can be assigned to tasks
- Keeps track of critical tasks
- Strong focus on visual representation of the project and its products
- Demonstrates potential success metrics [6]

4.2.2.4 Cons

The focus of this tool is very general to allow a full range of tools and techniques to be used when carrying out project management practises. Certain features that can be seen as negative points to this tool are listed below:

- Many features that might go unused
- Too much data needed for a small project
- To be fully effective external software is needed

4.2.2.5 Usefulness to this project

Overall this tool seems very fully featured, with similar functions to Microsoft Project but possibly easier to use. Some of these features that can be said to be useful to the tool being developed would be the ability to assign resources to tasks and how this tool tracks critical tasks. Aside from that this tool is very similar to other such tools.

4.2.3 FastTrack Schedule

4.2.3.1 Description

FastTrack Schedule (FTS) is a cross platform (available for Windows and Mac) project management software from AEC software. It is available in trial and purchasable versions and its website describes it as such,

> *"Your Windows project management software for organizing, tracking, and reporting all your project goals. Great for both new and experienced project managers, FastTrack Schedule 10 helps you manage projects easily and effectively." [7]*

4.2.3.2 Features

Being a consumer tool FTS is fully featured with many interesting and useful features, a list of these can be seen below:

- Plenty of template projects
- Different views (Gantt chart, calendar view and resource view)
- Can consolidate several project plans into a single overarching plan
- Seamless data exchange with MP [8]

4.2.3.3 Pros

There are several features that are very useful and can be seen as an advantage of using this tool and they are listed below:

- Cross platform
- Strong tutorial and example documents
- Thorough documentation.
- Can consolidate multiple project schedules into one document [8]

4.2.3.4 Cons

There are several things that this software does that could be improved on (potentially in the tool being designed for this project), these points are listed below.

- Possible to lose changes because the application doesn't prompt you to save on exit
- Inconsistent and sometimes awkward interface [8]
- Pricey
- A few interface quirks [9]

4.2.3.5 Usefulness to this project

Obviously the negative points of this software can be seen as useful areas to note down for things to look out for in this project there are also other useful things to take note of that may benefit this project, these are listed below.

- Thorough documentation [8]
- Expert-level control over projects
- Easy adjustments [10]

4.2.4 GanttProject

4.2.4.1 Description

The final project management software that shall be looked at is GanttProject (GP) which on its website is described as follows,

> *"GanttProject is a cross-platform desktop tool for project scheduling and management. It runs on Windows, Linux and MacOSX, it is free and its code is opensource."[11]*

This tool is different from the others above since it is open source and therefore caters to a slightly different audience than the rest.

4.2.4.2 Features

Despite being an open source project management tool GP is well rounded software with as many features as other competing products. Some notable features include.

- No need to download it to use it (online version)
- Resource assignment
- Milestone feature
- Saves files as .xml [12]

4.2.4.3 Pros

There are several key points that this software excels in and noting them down shall help build a picture of what all similar tools on the market currently have and should have. These points are listed below.

- Create work breakdown structure, draw dependencies, define milestones
- Generate PERT chart from Gantt chart
- Save charts as PNG images, generate PDF and HTML reports
- Import projects from and export them to Microsoft Project formats [13]

4.2.4.4 Cons

Certain areas are noted below for which GP could improve on and therefore areas that should be looked out for in this project when designing the tool.

- Only human resources available
- Reduced number of functions (due to nature of software)
- No integrated reporting
- Does not carry out calculations [14]

4.2.4.5 Usefulness to this project

A small amount of the features listed above that are good can be said to be useful to this project, these features are briefly listed below.

- Assign human resources to work on tasks

- Define milestones
- Resource load chart to view human resource allocation [13]

Overall this software is very useful to this project in terms of what it achieves as well as the handful of features it implements, since it is built using java and quite simple in what it tries to achieve.

4.3 **Conclusion**

The findings in this chapter are extremely important for this project, since they give a better understanding of the current environment for a tool such as the one being developed. Certain features found in the above tools that were either key to the user experience or were lacking, have been noted and shall be used when considering features needed and to be avoid for the tool being developed. Those key features are presented below.

- Gantt chart default view with different views possible
- Handle resources for a project
- Keeps track of critical tasks and milestones
- Automatic calculations
- Easy adjustments

Chapter 5 Analysis

Introduction

In this chapter both the functional and non-functional requirements for the tool presented and discussed in as much detail as possible to give an idea of how the tool should perform when in use. As previously mentioned in the previous chapter, there shall be several UML techniques that shall be used to illicit information and deal with them in such a way that the tool can go from inception to being built, all while following the defined methodology and corresponding lifecycle also mentioned in previous chapters. There was mention in above of functions that were extracted from given requirements these shall be used in this chapter to build the case study so that a better understanding of what is required and how the final tool shall be used by its users can be gained.

With the requirements laid out, modelling them according to the techniques previously mentioned shall begin with different levels of interaction by the users and how the final product shall work. For the sake of simplicity only some of the modelled requirements shall be shown in this document, those shall be;

- Adding a task
- Removing a task
- View a task
- Checking the priority of tasks

Models shall be shown in this report using all previously mentioned UML techniques appropriate for this chapter.

4.4 Background of the problem

At the moment students in the FoC have very little choice in what software to use for project management when carrying out their projects. Since most students prefer a simple easy to use yet fully functional tool but there currently is none, a tool that could fulfil such a position is much needed.

4.4.1 What are *project management tools*?

In the area of computing and computer science there are various ways to deal with a problem with many things to support the various aspects of such a problem. Usually when a problem is encountered and a decision has been made to deal with it then a project is started and various tools are used to support the project. Tools in this sense can come in many different forms therefore this project is mainly concerned with tools that are related to project management from a software engineering context.
Many definitions for project management tools can be found some of which are reliable and relevant to this project some of which are not. Below are definitions of what project management is and tools used to support project management.

According to PMBOK 2004 project management is:

> *"…the application of knowledge, skills, tools and techniques to project activities to meet project requirements. Project management is accomplished through the application and integration of the project management processes of initiating, planning, executing, monitoring, and controlling, and closing." [1]*

Additionally SWEBOK 2004 defines tools in the context of software engineering as such:

> *"Tools are often designed to support particular software engineering methods, reducing any administrative load associated with applying the method manually." [15]*

Therefore a project management tool would be designed to provide the support for the following key project management process:

- Initiating
- Planning
- Executing
- Monitoring and controlling
- Closing

Since this is a very broad area and topic, this particular project shall focus on the Planning and Monitoring and Controlling process of project management.

4.4.2 What does a *FoC student project* consist of?

As with real life projects a Faculty of Computing (FoC) student either undergraduate or post graduate student will have to undertake a project such as this one and go through "proper practises" (according to faculty and industrial standards) to complete said project.

Several things are needed when carrying out a project, these can be related to the individual carrying out the project and to the environment in which the project will be carried out. Specific requirements for carrying out a project from the point of view of a FoC student are as follows:

- Planning the project
- Researching the topic area
- Producing project documentation
- Producing a product

These requirements can relate to any project of any type but will always be part of a FoC student project along with potentially other requirements. The main area of interest for this project is area of planning the project and any related requirements to it.

4.5 **User Requirements**

4.5.1 Case Study

As mentioned above a case study shall be used to extract the requirements for the application that shall be created. This case study is based on observations of the current area, current tools and the conclusions drawn by the primary stakeholder of this project (the author of this document). Below the case study for this product is shown:

> A student who has started their project must work through the beginning of report and establish their requirements and how they shall manage their project. Once these details have been established then the student can pick out tasks that need to be done and feed these tasks into the tool giving each a start time and day and end time and day along with other information. The student may choose to either break down tasks into subtasks and input them into the tool or put the whole project start and end day and time into the tool. With the information above and

more entered into the tool a calendar view of the current month shall show which days have tasks assigned to them. A view by month, week and day may be selected and duration of each task shall be shown depending on what is selected as well as the priorities for each task (depending on their duration and possible dependencies between other tasks). As the students project goes further along tasks may be modified to accommodate for unforeseen or unexpected events, this shall be appropriately displayed on the tool.

4.6 **Functional requirements defined**

Below the functional requirements shall be defined by illustrating what happens (from the view of the tool and the user) when that function is run.

4.6.1 Add a task

Several text fields shall be presented to the user each asking for details of the task to be added, once each field has been filled in properly (e.g. correct data format) a button shall be pressed to save the task.

Input- Type in task details: name, date, description, deadline, dependencies, resources, mandatory, duration
Process- Check if data format for each task detail entered is correct then store
Output- Show new task on chosen start date

4.6.2 Modify a task

A task is chosen and its details shown all editable in a text field. Once user is done with modifying the task it shall be saved if the user wishes to do so.

Input- Type in task details: name, date, description, deadline, dependencies, resources, mandatory, duration
Process- Check if data format for each task detail entered is correct then store
Output- Show modified task on chosen start date

4.6.3 Delete a task

The user shall be able to delete a task by choosing from tasks already saved then deleting it.

Input- Choose a task by its name and (start) date.
Process- Identify the task by the given details above and remove the entry
Output- The project calendar shall display all entries but the one removed

4.6.4 View a task

The user shall select a day that contains tasks and shall select a view of all tasks that start on that day.

Input- Choose a day with tasks that start on that day and choose a daily view
Process- Identify the day and all tasks attached to the day
Output- All tasks that start on that day with their task details

4.6.5 Prioritise tasks

The user shall choose to view a list of tasks in order of priority; tasks that depend on other tasks shall have a lower priority than those that they depend on and tasks that have a closer end day/time shall have the highest priority.

Input- Task name, end date/time and dependencies
Process- Identify dependencies and which tasks are due sooner and order in terms of dependencies then due date.
Output- All tasks for the current project listed in order of priority

4.7 **Non-Functional requirement defined**

As well as the functional requirements, non-functional requirements shall be considered and defined to have a broader understanding of what is required of the tool. What follows is a detailed definition of the non-functional requirements, how they will be achieved and how the user might know that they have been achieved.

4.7.1 Configuration management

The product must follow basic configuration management principles, such as version control, uniform layout for the report and naming conventions for tool and report.

How it will be achieved

A template for writing the project report shall be used and followed with naming conventions established for both the report and the tool. Version control shall be maintained by having saving the report after key changes with a different version number and file name, and similarly for the tool (new implemented increments being taken as the key changes).

How the user will know that it has been achieved

There will be several versions of the tool (and report) each clearly labelled (date of changes, author and for the tool an incremental version number).

4.7.2 Platform compatibility

The tool shall ensure compatibility with all platforms that have the latest version of java and that are capable of running such java programs.

How it will be achieved

Writing the tool in the java programming language and testing on different platforms the tool will be said to be sufficiently compatible.

How the user will know that it has been achieved

By successfully running and using the tool on at least 3 different platforms that have the minimum system requirements and applicable environment for the tool, it can be said that platform compatibility has been achieved.

4.7.3 Maintainability

The tool shall be used frequently and should be able to have new features incorporated into it as the needs and requirements of the tool change, therefore the tool should be maintainable enough to allow such future changes.

How it will be achieved

By laying out source code in an easy to read fashion, commenting as much of the source code as possible and providing clear documentation of the how the tool works.

How the user will know that it has been achieved

If the maintainer spends more time making changes than trying to understand the code itself and if the number of desirable changes made is equal to the amount of initial desired changes.

4.7.4 Usability

The user should be able to use the tool intuitively to get what they require from the tool, and if any errors are faced then the user should be able to deal with the error(s) with little to no trouble.

How it will be achieved

By designing a user friendly, attractive interface with self explanatory text fields the usability can be maintained. Also this can be achieved by using clear and clean graphical elements on the user interface and error messages which are explained clearly so as not to confuse or frustrate the user.

How the user will know that it has been achieved

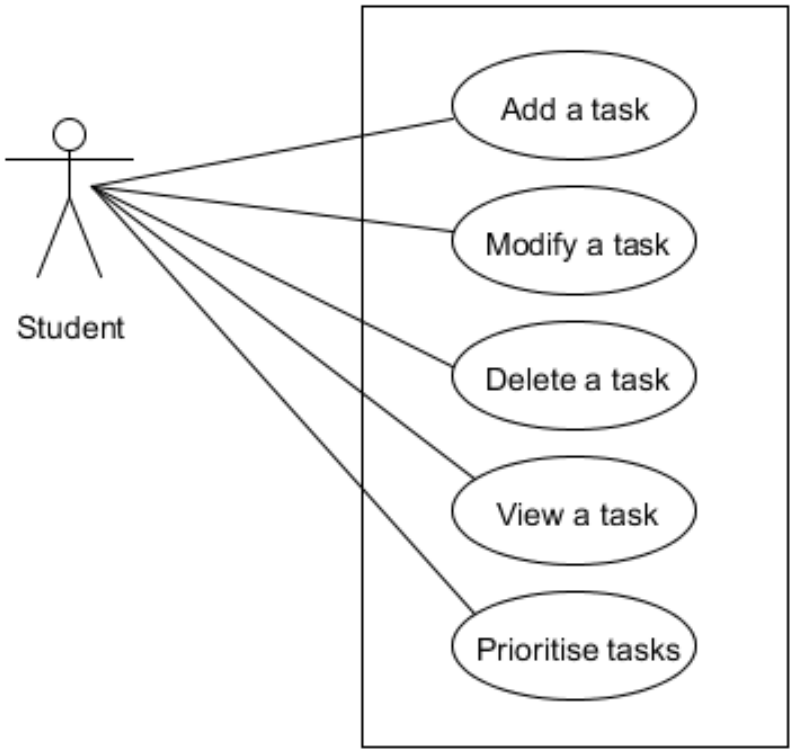
If the user can successfully use all features of the tool with only the provided documentation on their first time then it can be said the usability is high.

4.8 Use Case model

4.8.1 Description

With the functional and non-functional requirements gathered and detailed above the UML techniques for analysis can begin. The first of these UML techniques is the use case which is shown below.

4.8.2 Diagram: use case



This diagram shows a simplified view of the environment (although there may be many users and use cases, they will not be used or be useful to this project therefore they are not considered in this diagram and subsequent diagrams). Since there is only one user of the tool (the student) there is only one actor in the use case diagram.

4.9 **High level use case description**

4.9.1 Add a task

Actor: Student

Type: Primary

Description: The student runs the tool and is presented with the default calendar view which contains various text fields for entering a new task. Once the user has entered all the details for a new task they then click on the button to save the task which is added to the project calendar providing all mandatory fields have been filled in with the correct data format.

4.9.2 View a task

Actor: Student

Type: Primary

Description: The student runs the tool and is presented with the default calendar view (on the current month) and all tasks on that month. The users chooses a day that contains tasks that start on that day, then chooses a daily view which then changes the view to a list of tasks on the chosen day. If there are no tasks on that day then the view will be blank save for the date and giving the user the choice to look up tasks on another day.

4.9.3 Delete a task

Actor: Student

Type: Primary

Description: The student runs the tool then finds the task that shall be deleted by selecting its start date then viewing the tasks on that day. The name of the task to be deleted is entered then delete is chosen and the specified task is removed from the current project calendar.

4.9.4 Prioritise a task

Actor: Student

Type: Primary

Description: The student runs the tool making sure that there are tasks for the current project calendar. If all tasks have yet to be completed then the student shall click on the prioritise button then a list of all tasks shall be shown in order of priority (calculated by taking into consideration the due date, dependencies and mandatory fields of each task).

4.10 **Expanded use case description**

4.10.1 Add a task

Actor: Student

Type: Primary

Description: The student runs the tool and is presented with the default calendar view which contains various text fields for entering a new task. Once the user has entered all

the details for a new task they then click on the button to save the task which is added to the project calendar providing all mandatory fields have been filled in with the correct data format.

Dialogue:

Actor	Project Management tool
1) The student runs the tool then fills in the task details and then clicks the add task button.	
	2) The tool checks entered data to see if anything is missing or if the data format is wrong
	3) Displays confirmation of added task.

Alternative:

3) Displays an error message saying what needs to be changed; wrong data format or missing task details.

4.10.2 View a task

Actor: Student

Type: Primary

Description: The student runs the tool and is presented with the default calendar view (on the current month) and all tasks on that month. The users chooses a day that contains tasks that start on that day, then chooses a daily view which then changes the view to a list of tasks on the chosen day. If there are no tasks on that day then the view will be blank save for the date and giving the user the choice to look up tasks on another day.

Dialogue:

Actor	Configuration Management tool
1) The student runs the tool then selects a day that contains tasks and chooses daily calendar view.	
	2) The tool checks all tasks that start on that day.

	3) The tool displays all tasks starting on that day and their details.

Alternative:

3) If there are no tasks on that day then no tasks are displayed.

4.10.3 Delete a task

Actor: Student

Type: Primary

Description: The student runs the tool then finds the task that shall be deleted by selecting its start date then viewing the tasks on that day. The name of the task to be deleted is entered then delete is chosen and the specified task is removed from the current project calendar.

Dialogue:

Actor	Configuration Management tool
1) The student runs the tool then selects a day that contains tasks and chooses daily calendar view.	
	2) The tool checks all tasks that start on that day.
	3) The tool displays all tasks starting on that day and their details.
4) The user types the name of the task to be removed then clicks on the remove button.	
	5) The task gets removed from the project calendar and displays a confirmation message

Alternative:

3) If there are no tasks on that day then no tasks are displayed.

5) If the wrong task name is entered then an error message indicated that no such task exists is displayed.

4.10.4 Prioritise a task

Actor: Student

Type: Primary

Description: The student runs the tool making sure that there are tasks for the current project calendar. If all tasks have yet to be completed then the student shall click on the prioritise button then a list of all tasks shall be shown in order of priority (calculated by taking into consideration the due date, dependencies and mandatory fields of each task).

Dialogue:

Actor	Configuration Management tool
1) The student runs the tool then clicks on the prioritise button.	
	2) The tool checks all tasks by end time and dependencies.
	3) The tool displays a list of tasks in order of priority.

Alternative:

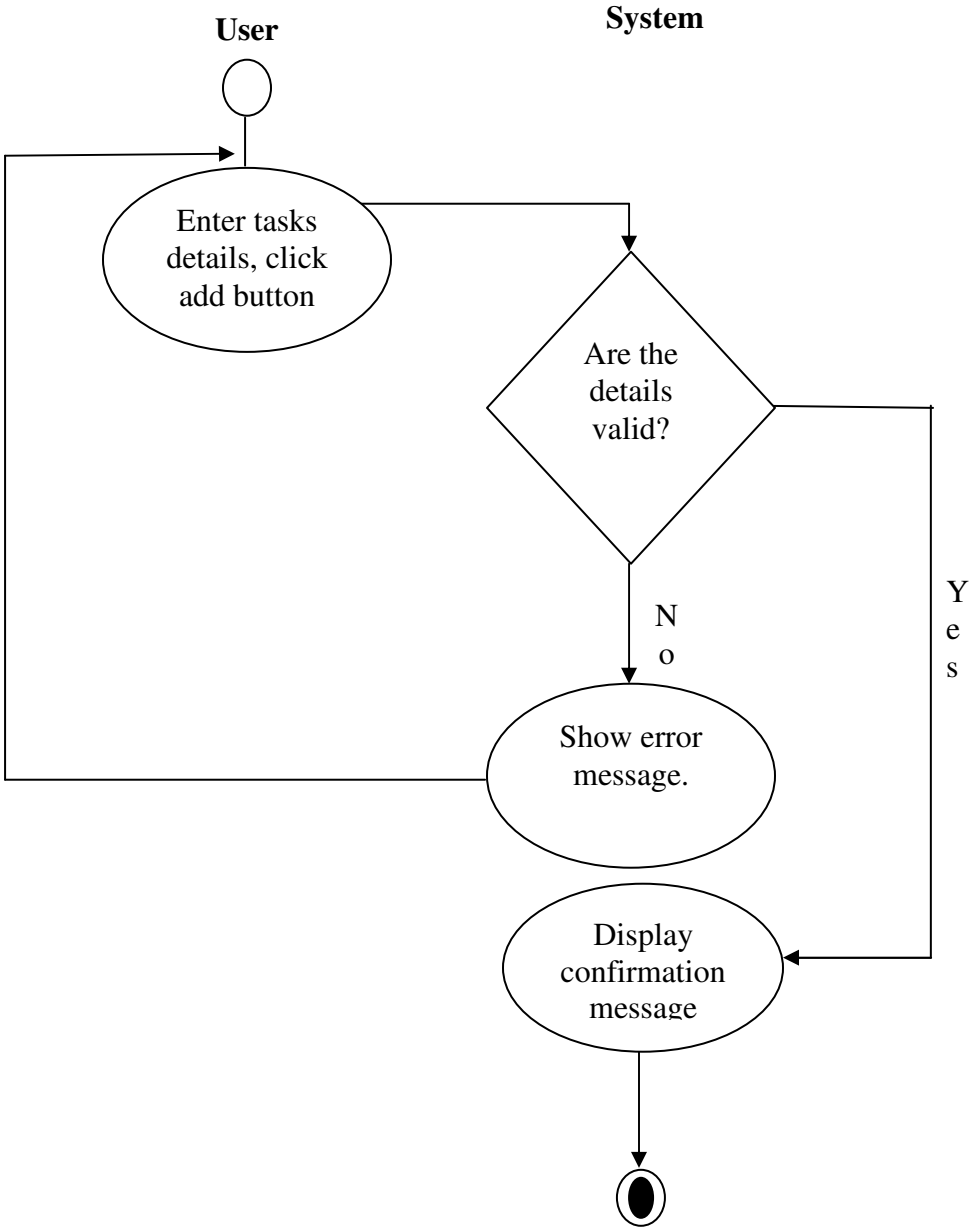
3) If there are no tasks then a message is displayed indicating that no tasks are available to prioritise.

4.11 **Activity diagrams**

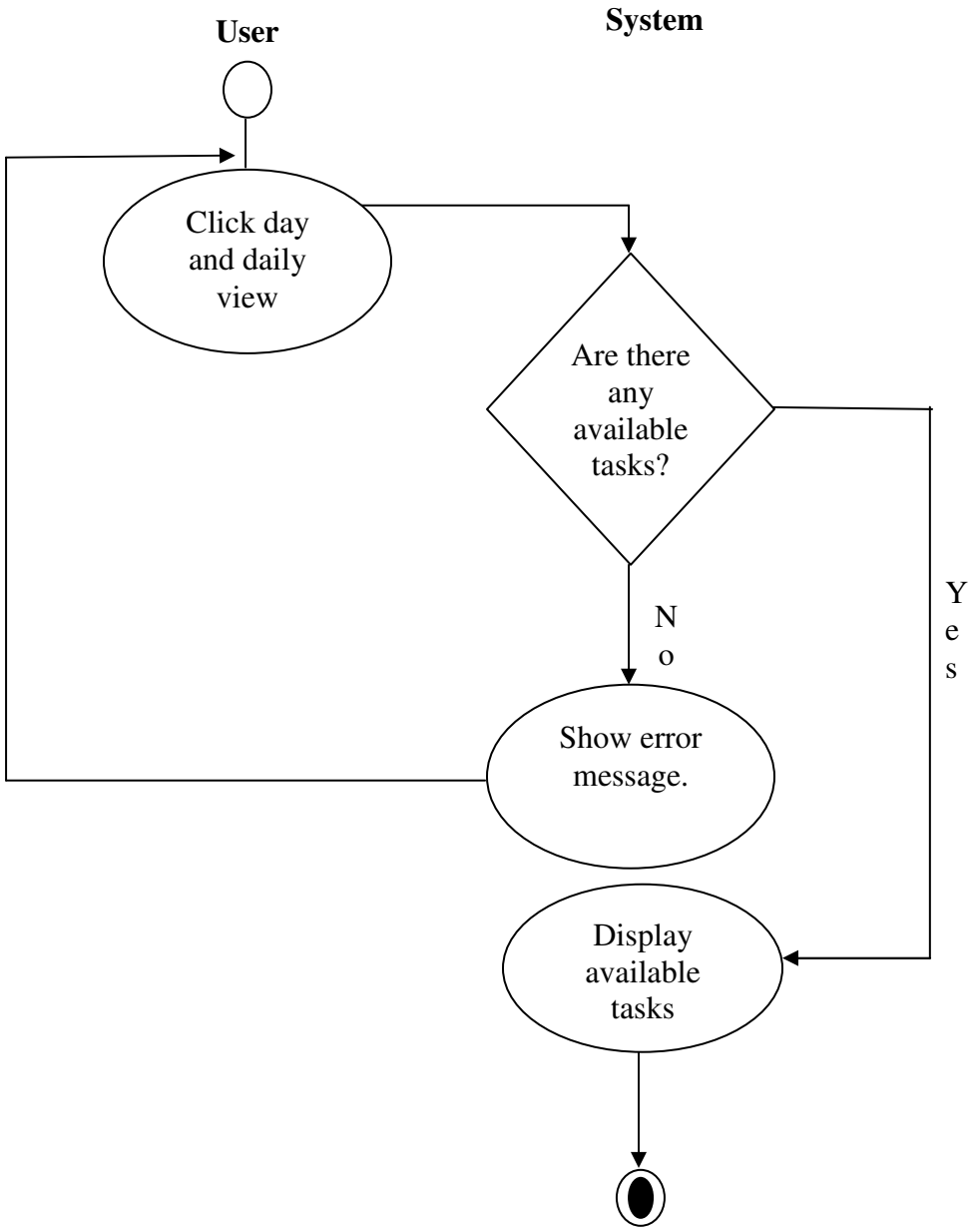
4.11.1 Description

With the use cases established for the product the information gained from them can be used to move on to the next UML technique by feeding the above data from each use case expanded description to their associated activity diagram which shall be modelled below.

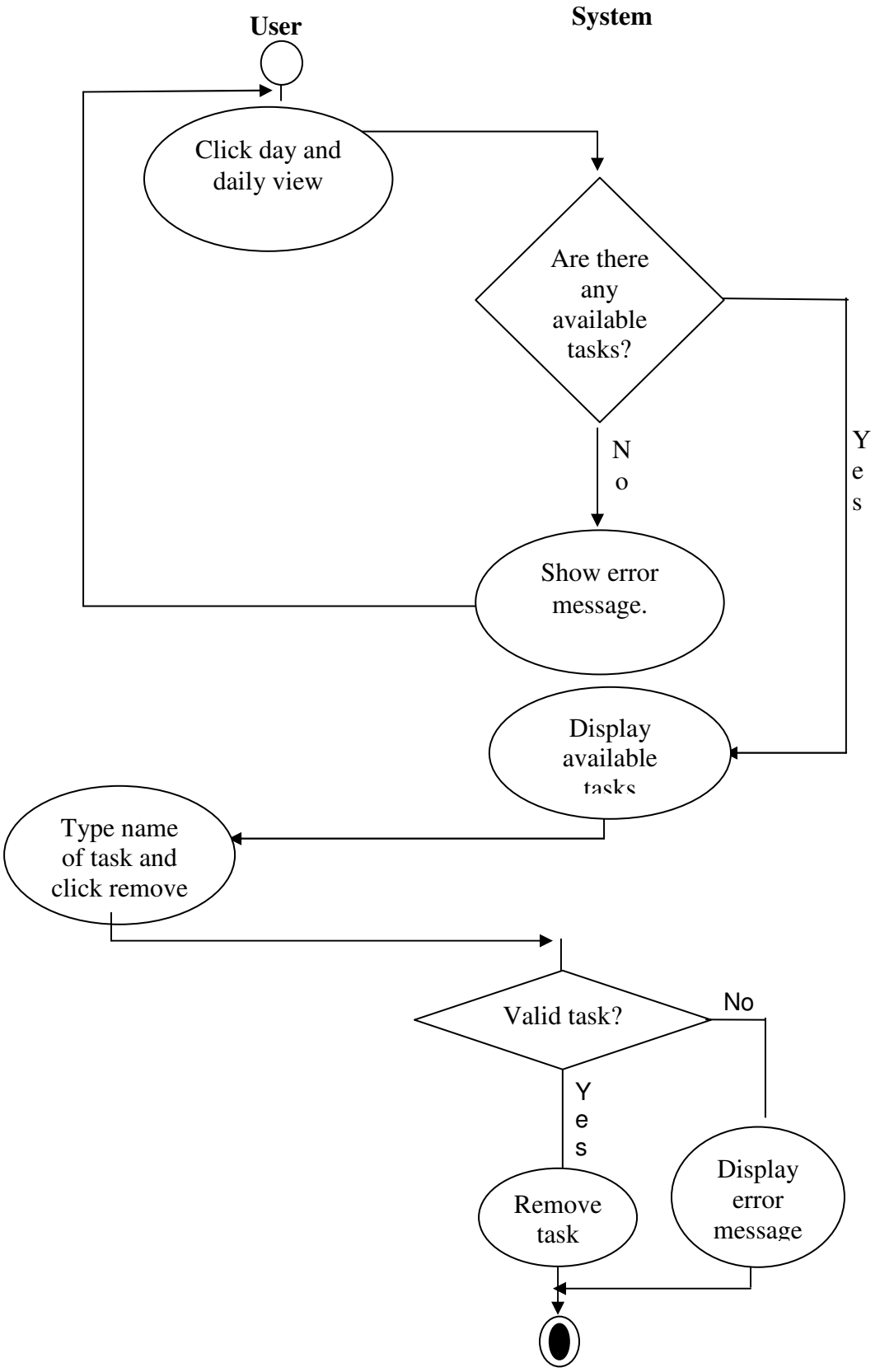
4.11.2 Diagram: Add a task



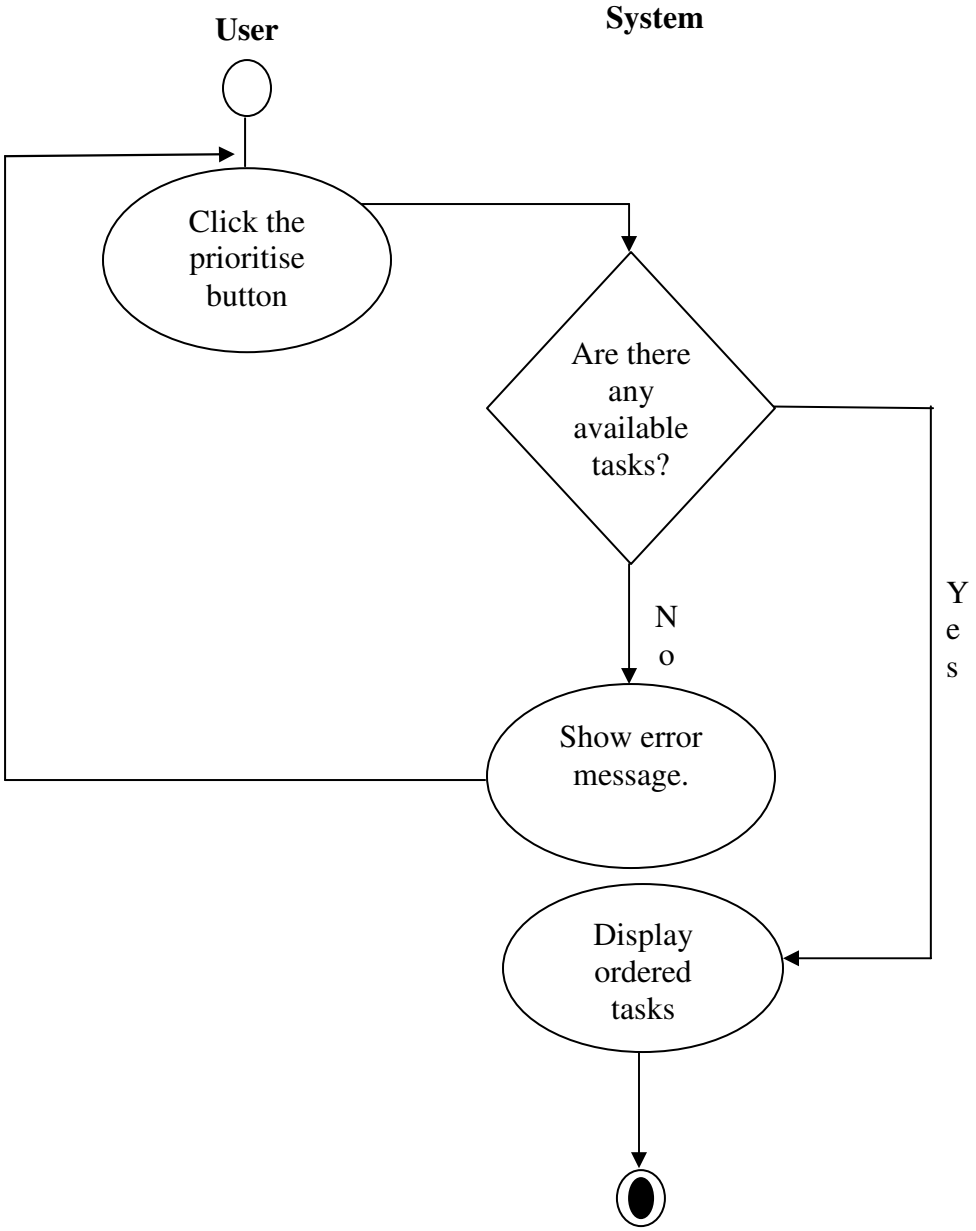
4.11.3 Diagram: View a task



4.11.4 Diagram: Delete a task



4.11.5 Diagram: Prioritise a task

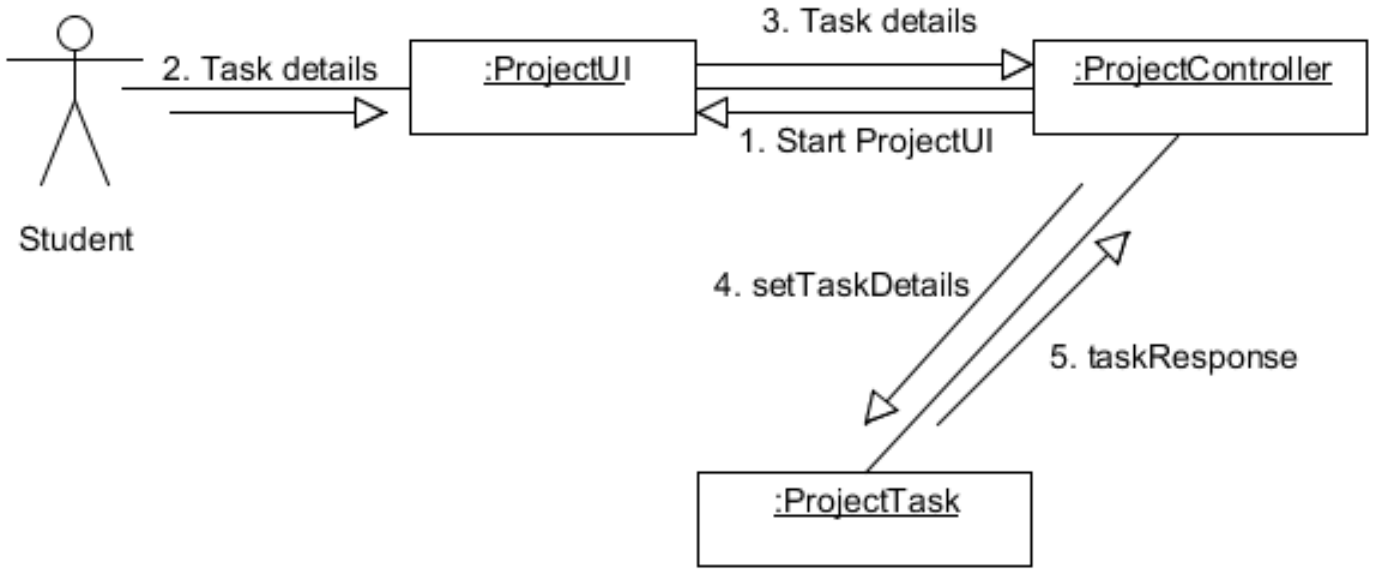


4.12 **Collaboration diagram**

4.12.1 Description

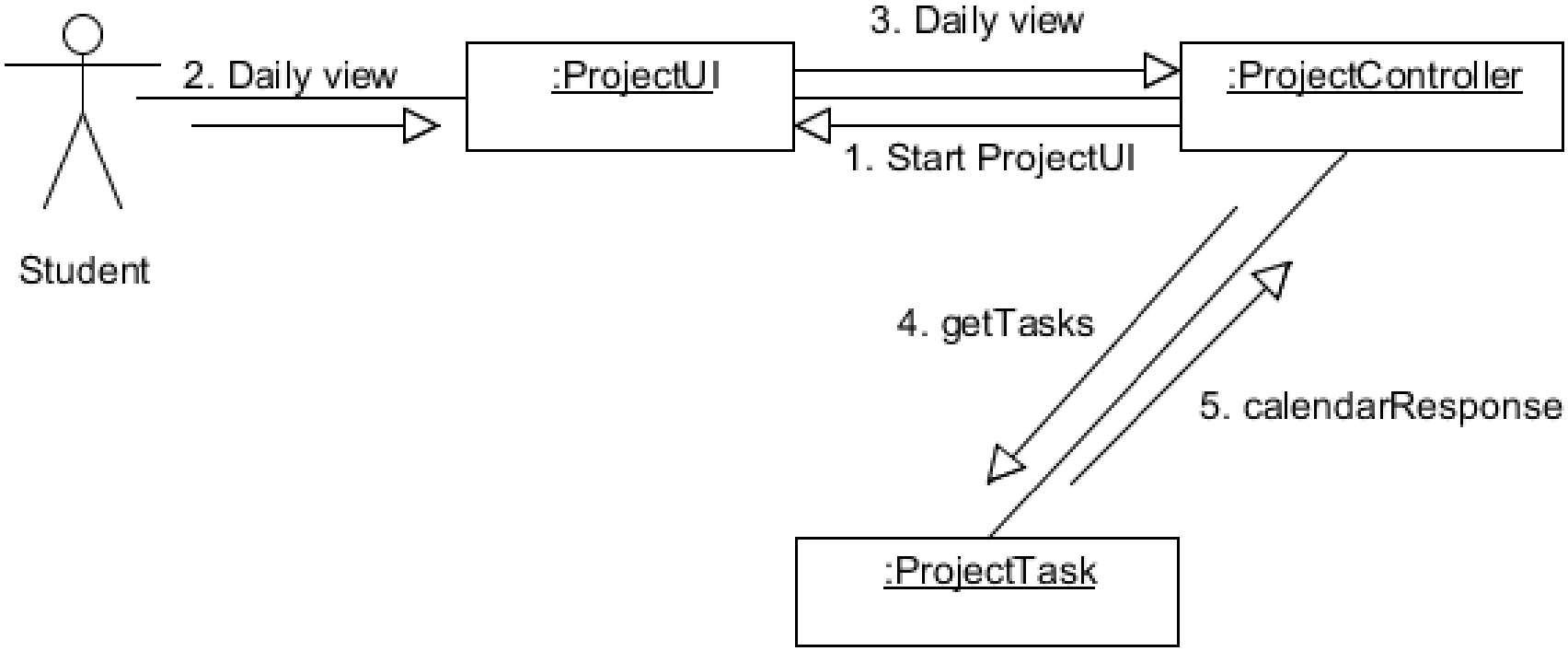
With the information gathered from both the use cases (detailed and expanded) as well as the activity diagrams the next step can be taken to transform the information into a more detailed logical form, as seen in collaboration diagrams which is shown below.

4.12.2 Diagram: Add a task



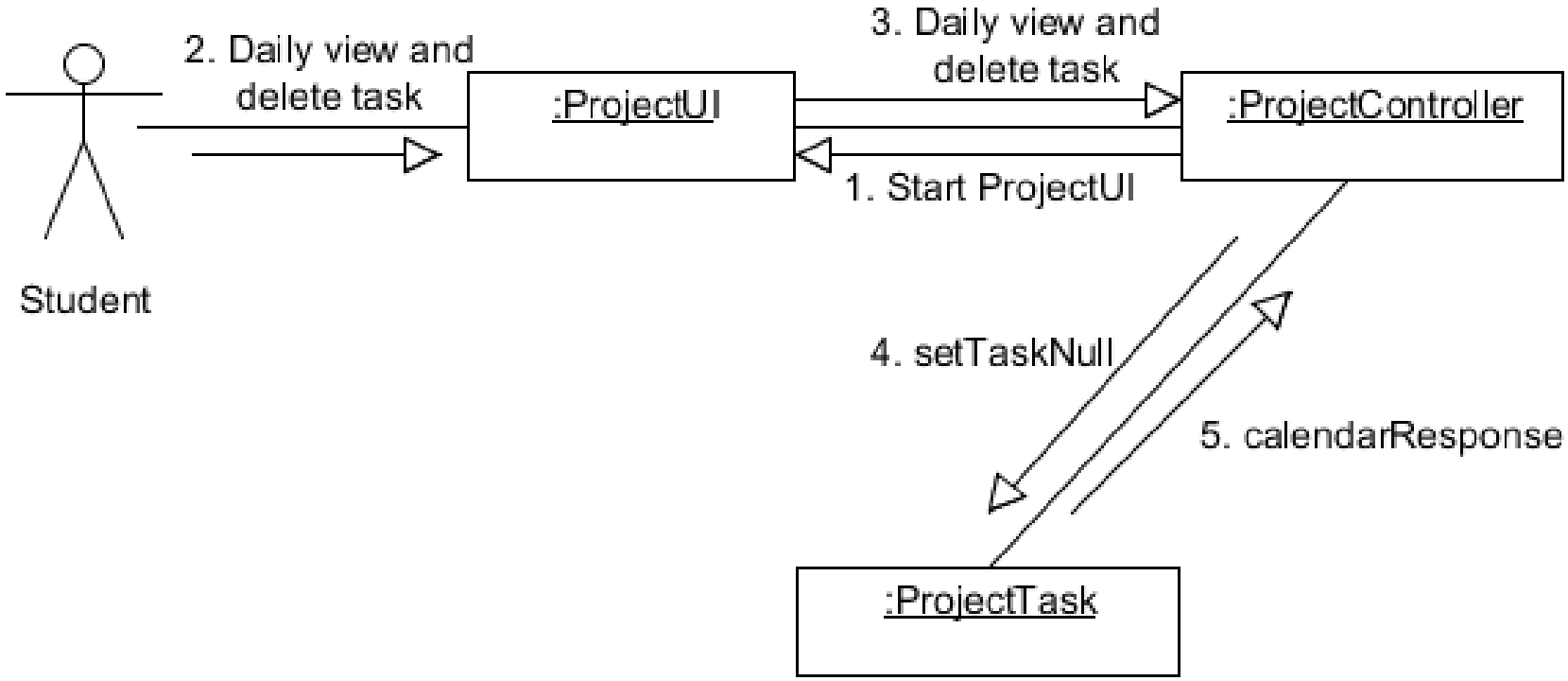
The numbers above represent the order of commands sent across each object and the user. In this case number the command, “5. taskResponse” is the tool informing the user of either a successful addition to the project or of an error that has occurred.

4.12.3 Diagram: View a task



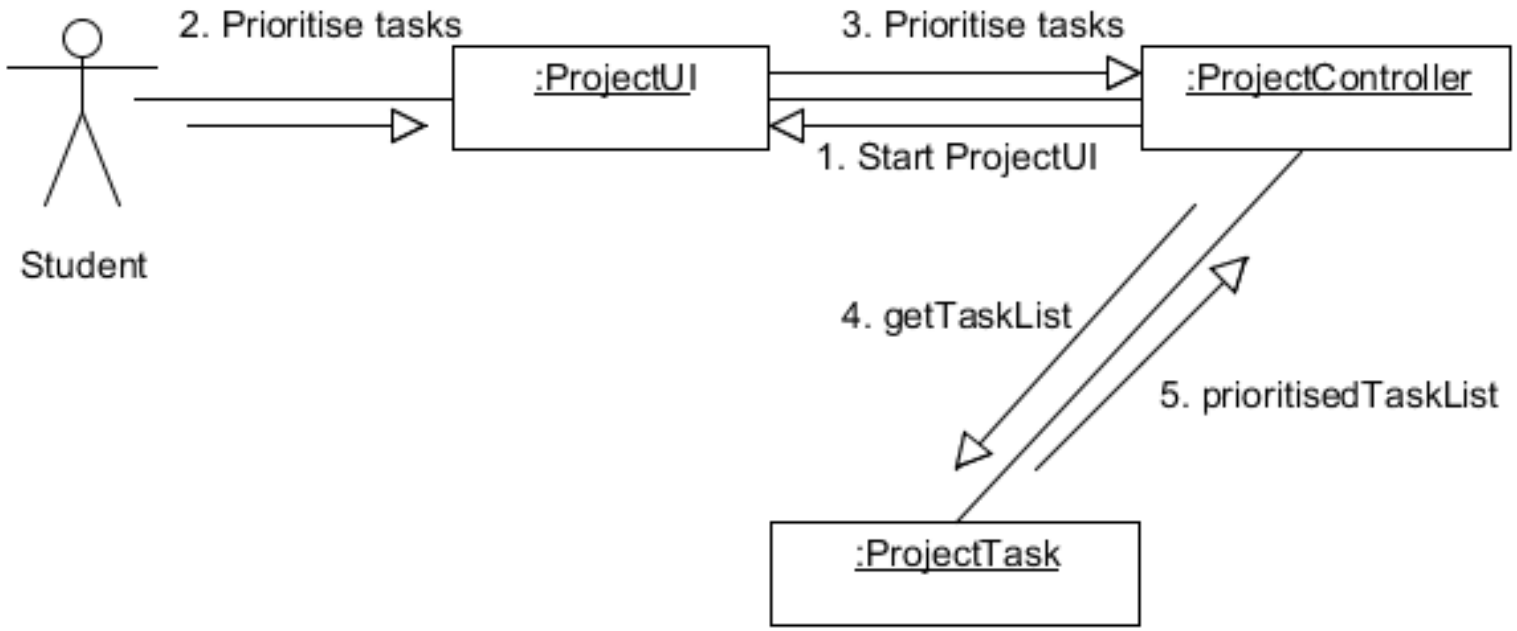
In this collaboration diagram the "Daily view" that the user selects is one of many choices of views (monthly, weekly and daily), but in this case it is used to show all tasks that start on the selected day. The "getTasks" response looks for all tasks on the selected day and the calendar response returns all tasks on that start on that day.

4.12.4 Diagram: Delete a task



This collaboration diagram shows that once the correct view and available task has been selected for deletion the task wipes all task details by setting values to empty (or null) as shown in “4.setTaskNull”, once again the “calendarResponse” is a message either displaying any possible errors that occurred or that the task has been successfully deleted.

4.12.5 Diagram: Prioritise a task



For this collaboration diagram the task list is being retrieved which shall be shown to the user in the calculated order of priority, the "prioritisedTaskList" being that list.

5. Chapter 6 Design

Introduction

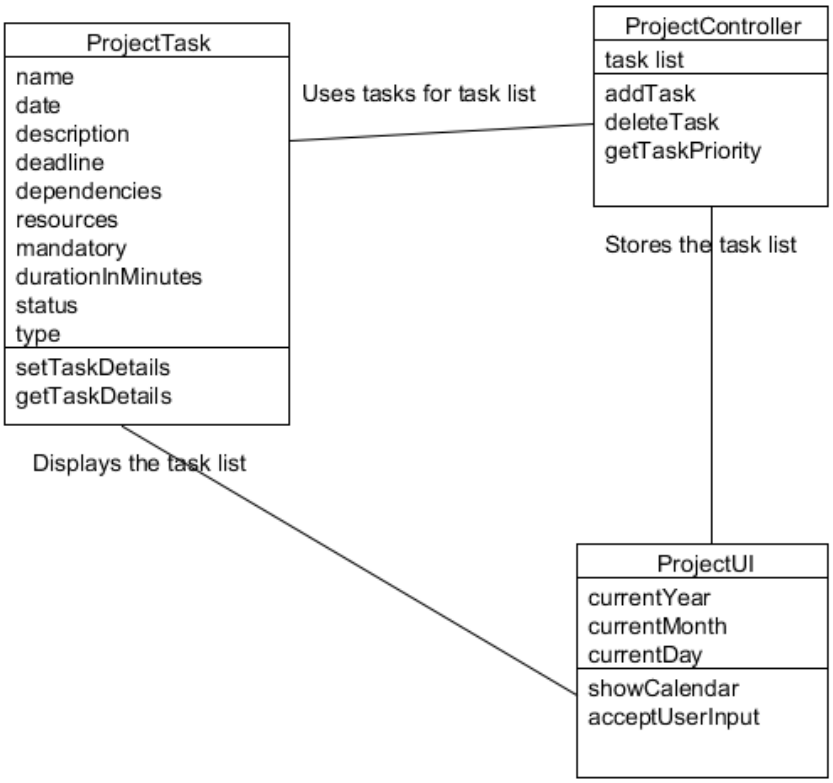
The focus of this chapter is to establish the logical and architectural design of the tool using the information gained from the analysis stage, also tools necessary for the later construction of the software. Designs of the user interface and how the interaction between the software and the user, shall be shown too along with software elements interfaces.

5.1 Class diagram

5.1.1 Description

All the information gained from the analysis stage, specifically the collaboration diagrams shall help model the class diagram which shall be integral to understanding the actual design of the tool and consequently its construction. The model presented below is the final version of the class diagram with all classes and relevant attributes and methods.

5.1.2 Diagram



5.2 **Data dictionary**

In this section the data gathered from the models in the analysis stage and from the class diagram shown above, shall be listed in the following data dictionary and explained in some detail.

Name	Alias	Where used/how used	Content description	Supplementary information
Name	taskName	Identify tasks(input)	Name=(String)	None
Date	startDate	Place task on calendar(output)	Date =(dd/mm/yyyy hh:mm)	None
Description	taskDescription	Identify more details to task(output)	Description=(String)	None

Deadline	None	Calculate priority(output)	Deadline=(dd/mm/yyyy hh:mm)	None
Dependencies	None	Calculate priority(output)	Dependencies=name	Takes the value of the names of tasks
Resources	None	Identify resources for tasks(output)	Resources=(String)	In other tool versions could be used for priority calculations
Mandatory	None	Check if task needs to be done(input)	Mandatory=True/False	In other tool versions could be used for priority calculations
durationinminutes	None	To show how long a task takes(output)	DurationInMinutes=Date-Deadline	Calculated by the tool
Status	None	Identify completed tasks(output)	Status=true/false	To check if a task has been completed or not
Type	taskType	Identify completed tasks(output)	Type="Task"/"Subtask"	Shows which are tasks and which are subtasks
taskList	None	Display all tasks(output) Identify tasks for prioritisation(input)	taskList=(task details)	All of the above used here as "task details"
currentYear	None	Display calendar(output)	currentYear=yyyy	System year used

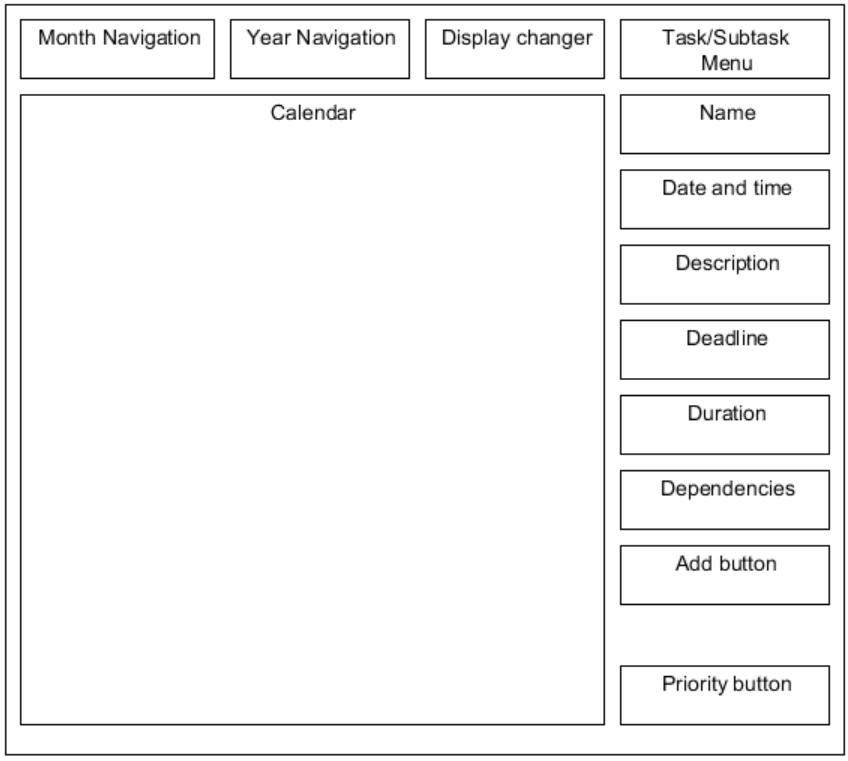
		Calculate date operations		
currentMonth	None	Display calendar(output) Calculate date operations	currentYear=MM	System month used
currentDay	None	Display calendar(output) Calculate date operations	currentYear=dd	System day used

5.3 **User Interface design**

In this section the main interface for the application shall be designed with an appropriate level of detail. A story board showing the main user interface of the application shall be shown below to better illustrate how the final tool shall look like.

5.3.1 Story board

Below is a representation (story board) of what the actual tool should look like while in use. Each section is labelled with the corresponding content and name appropriate for that section.



The monthly navigation area in the image above shall allow the user to choose from the previous month or next month to be displayed in the "Calendar" area below, similarly the year navigation area above does the same thing. The display changer is used to view the calendar in either yearly, monthly or daily view. Task/Subtask menu is used to choose what to add to the current project and using the spaces below to enter the task details then pressing the "Add button" to actually add the task. The priority button can be used at any time to check the priority of the stored tasks.

6. Chapter 7 Construction and testing

Introduction

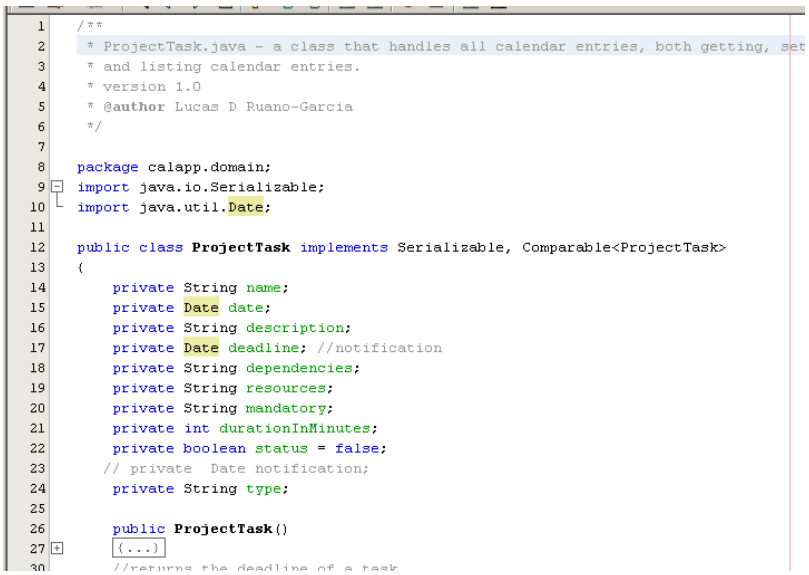
This core focus of this chapter is to document the construction process of the tool by using all the information that has been gathered in the previous chapters and with the aid of the designs given above. Construction however shall not be the only focus of this chapter as testing of the tool is extremely important while constructing the tool therefore some testing criteria shall be set and used as a test plan for the tool during and after construction.

6.1 Construction

The construction phase shall have one main product that shall implement all the required functions that user would want (using the gathered requirements and other information given in previous chapters). Since the chosen development language shall be java and the development environment shall be NetBeans, several screenshots shall be shown explaining each relevant part of the applications construction. Since certain classes shall rely on other classes for proper functionality, there shall be an order of which class is implemented first.

6.1.1 ProjectTask

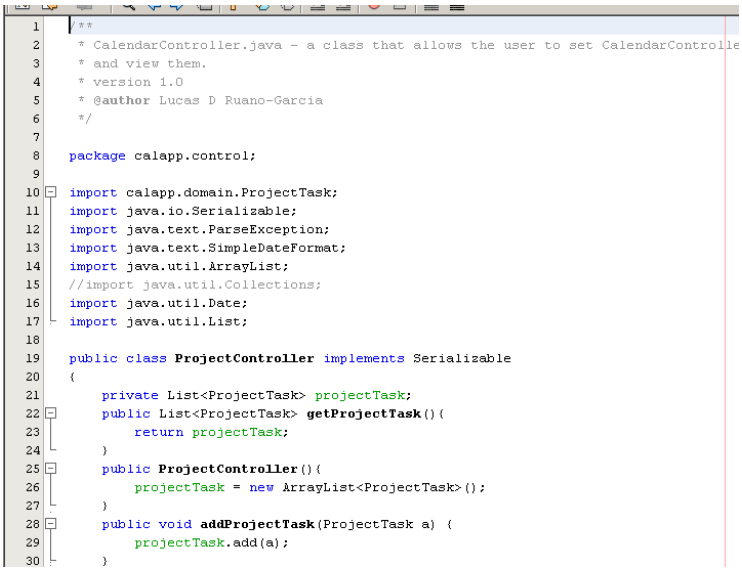
This class is at the heart of the workings of the tool, it holds all details of tasks which are then passed on to the other classes for them to work properly. Since this class essentially supports the other tasks by giving the needed information there is no main method. In creating this class the attributes were collected from the diagrams above and their types and other details were considered then entered into the beginning of the class. Also to allow the tool to be opened and closed without loosing information the class was made Serializable (the class that controls the GUI shall deal with storing to a file, opening and closing all details). Below is a screenshot of the beginning of the class during construction is shown.



```
/**
 * ProjectTask.java - a class that handles all calendar entries, both getting, set
 * and listing calendar entries.
 * version 1.0
 * @author Lucas D Ruano-Garcia
 */

package calapp.domain;
import java.io.Serializable;
import java.util.Date;

public class ProjectTask implements Serializable, Comparable<ProjectTask>
{
    private String name;
    private Date date;
    private String description;
    private Date deadline; //notification
    private String dependencies;
    private String resources;
    private String mandatory;
    private int durationInMinutes;
    private boolean status = false;
   // private  Date notification;
    private String type;

    public ProjectTask()
    {...}
    //returns the deadline of a task
```

6.1.2 ProjectController

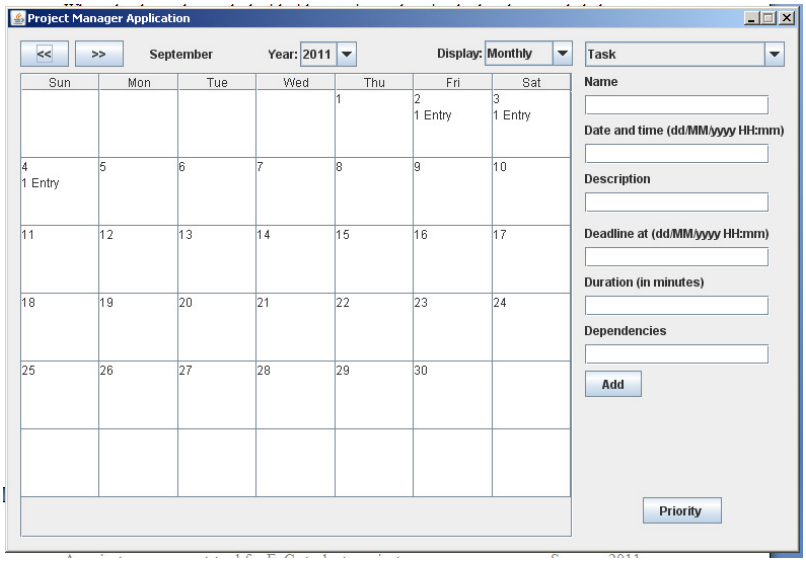
This class is essential to the proper running of the tool since it is the one that takes the tasks from the above class and stores it in an array list therefore adding tasks to the project and also removing tasks from the project. Again this class does not have a main method as above since once again it is supporting another class (the class that contains the user interface), therefore it cannot be run also it is Serializable. The information gained from the previous chapters was once again used to create this class with the help of the models and designs. A screenshot of the source code showing beginning of this class during construction is shown below.



```
/**
 * CalendarController.java - a class that allows the user to set CalendarControlle
 * and view them.
 * version 1.0
 * @author Lucas D Ruano-Garcia
 */

package calapp.control;

import calapp.domain.ProjectTask;
import java.io.Serializable;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
//import java.util.Collections;
import java.util.Date;
import java.util.List;

public class ProjectController implements Serializable
{
    private List<ProjectTask> projectTask;
    public List<ProjectTask> getProjectTask(){
        return projectTask;
    }
    public ProjectController(){
        projectTask = new ArrayList<ProjectTask>();
    }
    public void addProjectTask(ProjectTask a) {
        projectTask.add(a);
    }
```

6.1.3 ProjectFrame

Where the above classes deal with either getting and storing both tasks as a whole but also task details, this class deals with the GUI elements of the tool as well as tying the whole thing together for the user to run the tool and be able to use it properly. Since this class relies on the other two classes above it was created last, also since it deals with GUI elements of the tool it is much larger and therefore more time consuming to create, however NetBeans provided helpful in some cases for creating certain things. This class also implements the save to file (called "project.dat") using object output stream to write to the file and object input stream to read from the file, all done on running (retrieving from file) and closing (saving to file) of the tool. A screenshot of the tool running during construction is given below.



6.2 **Testing**

6.2.1 Purpose of testing

The aim of carrying out thorough testing on the application is to identify as many possible bugs and/or errors that maybe present as well as making sure that the requirements have been met and maintaining a high quality (as defined in previous chapters) product.

6.2.2 Testing Strategy

As mentioned above three functions shall be chosen to serve as samples of the application and with them their behaviour during testing shall be tested comparing the expected result (given in previous chapters) with the actual results achieved. For the testing to be carried out successfully some dummy data will be needed, these shall be as follows:

	Data set 1	**Data set 2**	**Data set 3**
Name	name	Task1	Task2
Date and time	4/9/2011 10:00	4/10/2011 10:00	10/04/2012 10:00
Description	desc1	Desc2	Desc3
Deadline	4/12/2011 10:00	4/12/2011 12:00	04/05/2012 12:00
Duration	10	5	20
Dependencies		Task2	

6.3 **Test Case's**

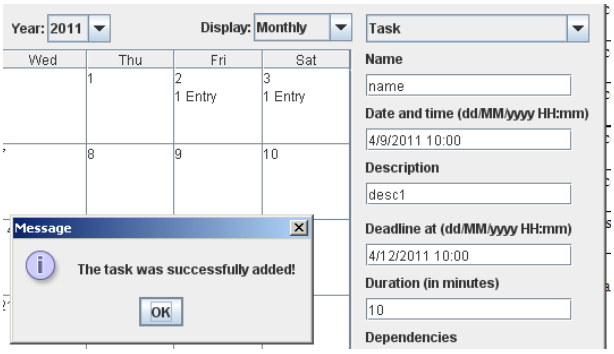
Test Case: Add a task

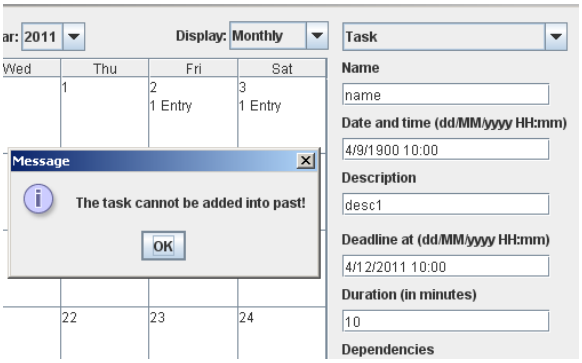
Test Case Id	**Input values**	**Test field name**	**Expected Result**	**Actual Result**	**Comments**
1	4/9/2011 10:00	Date and time	Successfully add to project	Added with no problems	Successful
	4/12/2011 10:00	Deadline	Successfully add to project	Added with no problems	Successful
	Task1	Dependencies	Successfully add to project	Added with no problems	Successful
2	4/9/1900 10:00	Date and time	Error message and not added	Error message and not added	Successful
	4/9/1900 10:00	Deadline	Error message and not added	Error message and not added	Successful
	(blank)	Dependencies	Successfully	Added with no	Successful

			add to project	problems	
3	abc	Date and time	Error message and not added	Data format error message	Successful
	Abc	Deadline	Error message and not added	Data format error message	Successful
	noTask	Dependencies	Successfully add to project	Added wrongly	Unsuccessful

From the three fields tested it can be said that they are working very well and that overall this test has been a success with all expected results being realised, except for the Dependencies field which accepted a nonexistent task, however this could be seen as a good thing (since the nonexistent task could be added afterwards) therefore not causing a huge mistake.

A sample of screenshots from the test results is given below

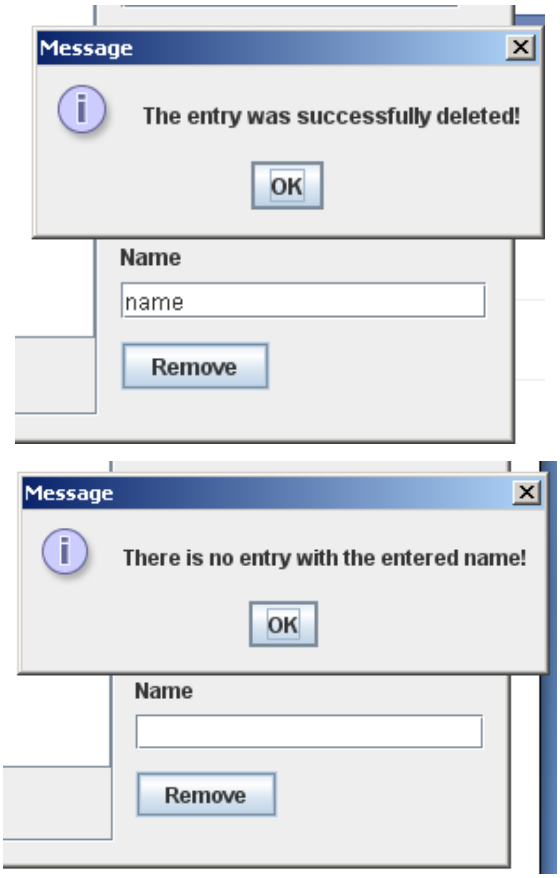




Test Case: Remove a task

Test Case Id	Input values	Expected Result	Actual Result	Comments
1	Name	Successfully removed from project	Removed with no problems	Successful
	Task1	Successfully removed from project	Removed with no problems	Successful
	Task2	Successfully removed from project	Removed with no problems	Successful
2	Task0	Error message and nothing removed	Error as expected	Successful
	Task3	Error message and nothing removed	Error as expected	Successful
	Task	Error message and nothing removed	Error as expected	Successful
3	(blank)	Error message and nothing removed	Error as expected	Successful
	123	Error message and nothing removed	Error as expected	Successful
	<>?	Error message and nothing removed	Error as expected	Successful

This test has been completely successful, all existing tasks were removed as expected and nonexistent tasks drew error messages from the tool saying that the task was not found therefore could not be deleted. Below are some screenshots of the results of this test case.



Test Case: Prioritise tasks

Test Case Id	Input values	Expected Result	Actual Result	Comments
1	(dummy data "name")	Should be first	First	Successful
	(dummy data "task1")	Should be second	Second	Successful
	(dummy data "task2")	Should be third	Third	Successful
2	New data1	Should be first	First	Successful
	New data2	Should be second	Second	Successful

	New data3	Should be third	Third	Successful
3	New data4	Should be second	First	Unsuccessful
	New data5	Should be first	Second	Unsuccessful
	New data6	Should be third	Third	Unsuccessful

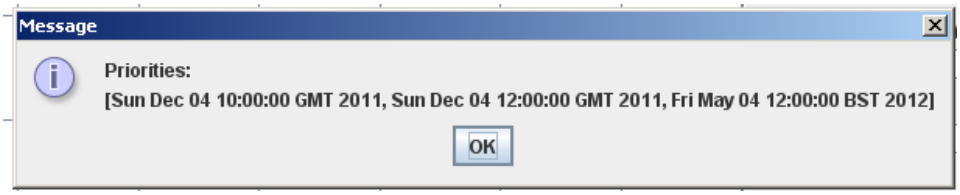
The data used for this test case is shown below in two separate tables.

	Data set 1	**Data set 2**	**Data set 3**
Name	New data1	New data2	New data3
Date and time	4/9/2011 10:00	4/10/2011 10:00	10/04/2012 10:00
Description	desc1	desc2	desc3
Deadline	04/12/2011 10:00	04/12/2011 12:00	04/12/2012 13:00
Duration	10	5	20
Dependencies			New data1

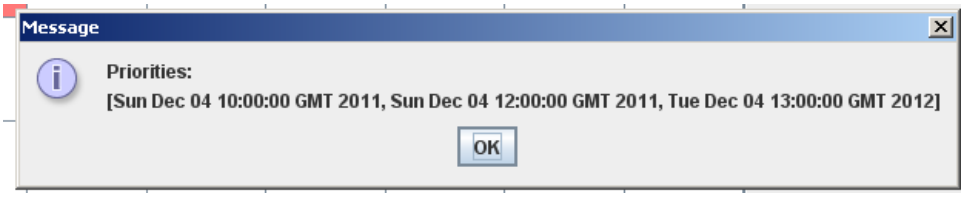
	Data set 1	**Data set 2**	**Data set 3**
Name	New data4	New data5	New data6
Date and time	4/9/2011 10:00	4/10/2011 10:00	10/04/2012 10:00
Description	desc1	desc2	desc3
Deadline	04/12/2011 10:00	04/12/2011 12:00	04/12/2012 13:00
Duration	10	5	20
Dependencies	New data5		

Overall this test has performed reasonably well, there were a few unexpected results in the test case with the id of 3, in that the priority order came out in an unexpected manner, this may be due to the fact that only the deadline is being taken into account when ordering the priority of each task, rather than taking the dependencies into account too as it should. This is a minor problem though that can be solved with more time. Below are screenshots of the results from this test case.

Successful result.



Unsuccessful result.



6.4 User Interface Testing

Several tests shall be carried out to test the user interface against what is expected. These tests are listed below:

- Consistency testing (covers the look of the tool)
- Button testing
- Menu testing

6.4.1 Consistency testing

The aim of this test is to determine whether the tool as a whole has a consistent look when using several of its features. The way this test shall be carried out is by navigating through different views and using functions that change the view of the tool and comparing the resulting views with the initial view seen when starting the tool. If the changes in views are dramatic then the test shall be said to not pass, if however the changes in view are unnoticeable or not very dramatic then the test shall be successful. Consistency is measure as follows: Consistent, quiet consistent and not consistent.

Views	Consistent layout?	Notes
Monthly	-	This is the baseline for which other views shall be compared to
Weekly	Consistent	Only the calendar area changes therefore the rest of the tools layout is left consistent.
Daily	Consistent	Only the calendar area changes therefore the rest of the tools layout is left consistent.
Day/Month changer	Consistent	If on monthly view changes are only made to calendar area and if on daily view again changes are made only to the calendar area.
Year changer	Consistent	Only the calendar area changes therefore the rest of the tools layout is left consistent.

In terms of consistency in the effects of each function manipulating the layout of the tool, all tests have passed, also over all the tool maintains a clear and clean view when changing from one view to another with all functions working as expected and views displayed as expected too.

6.4.2 Button testing

In this test the buttons of the tool shall be tested to see if they respond as expected, or if something else happens. This shall be documented below in a simple table.

Button name	Expected results	Actual results
Back (month/day)	Change the view to the previous month/day	View changed successfully
Forward	Change the view to the next	View changed successfully

(month/day)	month/day	
Add	(with all details filled in) Add a task to the project	Task added successfully and message displayed
Remove	(with all details filled in) Remove a task to the project	Task removed successfully and message displayed
Priority	(with data) Show priority of tasks	Priority of tasks displayed

All the buttons that were tested performed as expected with no notable errors or unexpected occurrences, therefore it can be said that these tests were successful and all buttons are working well.

6.4.3 Menu testing

This test is similar to the test above, the only difference is that the focus is on the menus of the tool (drop down menus). As above expected and actual behaviour of each menu shall be tested and noted in the table below.

Menu name	**Expected results**	**Actual results**
Year changer	List of years from 1911-2111 shall be shown in the drop down menu which when clicked shall show the appropriate view	Successful
Display changer	Monthly, daily and weekly view options shall be shown allowing the user to click on either to change the calendar view	Successful
Task type changer	Task and subtask shall be shown as a choice and allow the user to choose either so that one can be added to the project which shall then be shown when the task is viewed	Successful

This test performed as expected with no noticeable unexpected behaviour occurring.

6.5 **Testing Review**

This section closes the testing chapter looking at the overall results from the individual tests carried out above with a review of the overall performance of the application.

6.5.1 Pass & Failure rate

Out of all the tests carried out on the tool both on interface and functions, only a small portion of the tests performed unexpectedly, but as noted above the behaviours that were noted are not major and can either be fixed easily or seen as a potential benefit. It must be noted however that these results are from the sample data, meaning although the overall result is positive, it is possible that more tests would bring up more errors and consequently the possibility of more failures.

6.5.2 Errors and Solutions

There were a small portion of errors that were found during testing as mentioned above, these shall be discussed in more detail in this section along with possible solutions. The first error that was encountered was that the dependencies field accepted any input rather than an existing task, this could be a problem since a task should not depend on something that does not exist (it would not make sense and cause problems when calculating task priority). A possible solution to this would be to have the dependencies field read all task (names) and check if the input matches or not, however this is not a major issue since it can also serve handy when tasks are inputted in the wrong order or certain tasks are forgotten when adding to the project.

The other error that was encountered is a more serious error that involves calculating priority using the dependencies field and end date. It seems that the dependencies are not considered therefore allowing tasks to be prioritised wrongly, this could be a major problem since if a task is dependent on another it can not be done before it therefore the tool would not carry out a major function expected of it. A solution to this would be to check the function that is used to calculate the priority and review it to see if dependencies are being considered and if not then add them and if so then take a more in depth view of what exactly is causing the error.

6.5.3 Conclusion

The application as a whole carries out certain functions successfully with minor errors but these can solved in later versions.

7. Chapter 8 Conclusions and evaluation

Introduction

This chapter marks the closing of the project as a whole and therefore deals with any final thoughts about the project and the product developed and an evaluation of how the project and product have been carried out, with any possible improvements.

7.1 Conclusion

As the result of the tests that have been carried out show and the above evaluation many aspects of the product have been carried out well with few errors, however many things also could be improved aside from the above mentioned solutions to known errors there are more functions which would add a unique identity to this tool and make it stand out more among the other available tools.

In the analysis phase the background of the project was taken and a general idea of what the tool would need was gained. On the level of the project the report had started to take shape and project management principles were being used to keep track of the progress. Once the design phase had been reached then the requirements that had been gained previously were developed and a better understanding of how the tool would be internally and what it would need was reached. On the project level again the report had progressed well and a better understanding of how it should be carried out was gained. The construction and testing phase of the tool held many problems and therefore took the longest time, however after a while the tool started to gain shape and the functions that were expected started to work as expected. By that time the project was nearing closing and the report was more than half way. Further testing was done to check that the functions were all working as expected and the final chapters of the report were being written.

There have been many milestones in this project all of which have seemingly been met, in terms of achievements the fact that a tool that carried out (limited) functions described at the beginning of this project has to be the biggest of achievements.

7.2 **Evaluation**

Overall the project has been successful in implementing several of the desired features albeit with a few errors, however as a whole the tool is functional and can be used for its intended purpose, although given more time and effort features could be improved and additional functions could be added to enhance the user experience and improve project time management for students as a whole.

8. References

1. A Guide to the Project Management Body of Knowledge. Project Management Institute. 2004. Third edition.
2. ERCIM. “Terminology”. Internet: http://wiki.ercim.eu/wg/SoftwareEvolution/index.php/Terminology, [20/05/2011].
3. Association for Project Management. “Glossary”. Internet: www.apm.org.uk/Definitions.asp. [20/05/2011].
4. J. Paynter. “Software engineering project management, estimation and metrics: discussion summary and recommendations”. *International Conference Proceedings Software Engineering: Education and Practice*. 1996. pp. 500.
5. Microsoft. “Project Pro 2010 benefits”. Internet: http://www.microsoft.com/project/en/us/project-pro-2010-benefits.aspx [13/06/2011].
6. ConceptDraw. “ConceptDraw Project 6”. Internet: http://www.conceptdraw.com/products/conceptdraw-project-2. [13/06/2011].
7. AEC Software. “Fast Track 10 for Windows”. Internet: http://www.aecsoftware.com/project-management-software/fasttrack-schedule-win/ [13/06/2011].
8. M. Sarrel. “FastTrack Schedule 9.1 Review & Rating”. Internet: http://www.pcmag.com/article2/0,2817,2090751,00.asp [13/06/2011].
9. MacFormat. “AEC Software FastTrack Schedule 9 review”. Internet: http://www.techradar.com/reviews/pc-mac/software/business-and-finance-software/aec-software-fasttrack-schedule-9-32209/review [13/06/2011].
10. J. Brandon. “FastTrack Schedule 10 review”. Internet: http://www.macworld.co.uk/digitallifestyle/reviews/index.cfm?reviewid=3229132 [13/06/2011].
11. GanttProject Team. “GanttProject Home”. Internet: http://www.ganttproject.biz/ [14/06/2011].
12. J.-C. Cutter. "Gantt Project – Review". Internet: http://www.softwareprojects.org/reviews-gantt-project.htm [14/06/2011].

13. T. Brandstetter. "Review PM Software GanttProject". Internet: http://pm-software.org/reviews/review-pm-software-ganttproject [14/06/2011].
14. Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK). Project Management Institute. 2004. First edition.

9. Bibliography

1. R. S. Pressman. Software Engineering a practitioner's approach. McGraw Hill. 2001. Fifth edition.
2. T. Luckey, J. Phillips. Software Project Management for Dummies. Wiley Publishing Inc. 2006. First edition.
3. S. Walther, J. Levine. Teach Yourself E-Commerce Programming with ASP. SAMS. 2000.
4. B. Helbrough, "Computer assisted collaboration the fourth dimension of project management?", *International journal of project management :the journal of the International Project Management Association*, vol. 13, No. 5, pp.329-333, 1995.
5. J. Voelcker, "Automating software: proceed with caution", *IEEE Spectrum*, vol. 25, no. 7, pp. 25, 1988.
6. R. Fabac, et al., "Frequency of Use and Importance of Software Tools in Project Management Practice in Croatia", *32nd International Conference on Information Technology Interfaces (ITI)*, Cavtat/Dubrovnik, 2010, pp 465.
7. S. Goldmann, "Procura: a project management model of concurrent planning and design", *Proceedings of the 5th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Stanford, CA, 1996, pp 177.
8. M. Daneva, R. Terzieva, "Assessing the Potentials of CASE-Tools in Software Process Improvement: A Benchmarking Study", *Assessment of Software Tools, 1996., Proceedings of the Fourth International Symposium*, 1996.
9. AMI: Application of Metrics in Industry, Metrics Users' Handbook, AMI Consortium, 1992.
10. Ashman, R., "Project Estimation: A Simple Use-Case-Based Model", *IT Professional*, vol. 6, no. 4,. 2004.
11. P. W. Oman, "CASE: analysis and design tools", *Software, IEEE*, vol. 7, no. 3, pp. 37, 1990.
12. F. Kaplan, "Application of CASE tools and object oriented programming to the software development process", *IEEE International Conference on*

Communications, 1989. ICC '89, BOSTONICC/89. Conference record. 'World Prosperity Through Communications', Boston, MA, 1989, pp. 1319.

13. M. Chen, R. J. Norman, “Integrated computer-aided software engineering (CASE): adoption, implementation, and impacts”, *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences, 1992.*, Kauai, HI, 1992.
14. A. Yuval, “The integration of CASE and a software engineering framework”, *Fifth Israel Conference on Computer Systems and Software Engineering*, Herzlia, Israel, 1991.

10. Appendix

To run the project management tool and carry out various operations please follow the steps give bellow.

Opening and starting up the tool

1. Open the “dist” folder and click on “Project Manager.jar” file

Add task/subtask

1. Fill in all the fields to the right correctly (watching for date formats and required information)
2. Click “Add” button
3. Task should be added by start date

Remove task

1. Click on a day that has a task
2. Click the drop down menu next to “Display” and choose, “Daily”
3. In the “Name” field type the name of the task (as is given to the left)
4. Click “Remove” button and task should be deleted

Prioritise tasks

1. Ensuring there are tasks in the project click the button “Priority”
2. A message should pop up with the tasks ordered by priority